scalable Smart aNalytic APplication builder for sentient Cities: for Living Lab and co-working with Stakeholders

https://www.Snap4City.org

Data Ingestion Processes and Tools

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

Paolo Nesi, paolo.nesi@unifi.it
https://www.Km4City.org
https://www.disit.org
Snap4City/Industry structure

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

- **Https://www.Snap4City.ORG** is the main instance of Snap4xxxx solution managed by DISIT Lab. The main documentation is located and updated on Snap4City.org, GitHub, dockerHub and Node-Red Library. Snap4City.org is where the last tools are tested and news published.
  - Organizations on Snap4City.org have been created with contracts as for _Platform as a Service_, for testing and for providing _SmartCity as a Service_ as well as _Industry 4.0 as a Service_
Most of Organizations on Snap4City.org also correspond to companies or institutions that have an installation of Snap4City tools on their Premise,
   - such as: Pisa, SmartGarda Lake, Snap4, ALTAIR, etc.

This double way allows them to:
   - test the news,
   - share experiences with other groups,
   - get visibility,
   - work in the collaborative environment, and
   - be better supported by Snap4City.org and DISIT Lab personnel.

Each instance of Snap4xxxx solution can decide to join the federation of SmartCity API to exploit shared data.
   - This allows to exploit regional data for city installations applications (web, mobile, dashboards, etc.) without reloading them for example.

Main Organizations/areas
   - Antwerp area (Be)
   - Capeion (Sweden: Västerås, Eskilstuna, Karlstad)
   - DISIT demo (multiple)
   - Dubrovnik, Croatia
   - Firenze area (I)
   - Garda Lake area (I)
   - Helsinki area (Fn)
   - 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)
A Mobile App may refer to one Smart City API Server (for Area 1) via SUPER and receive data from the Federated SUPERS (Area 2) if navigation, queries, etc. are leading to discover out of the addressed KB.

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

- The usage of Super (ServiceMap) is not mandatory so that separate services can be produced as well.
- SuperServiceMap and ServiceMap presents the same Smart City APIs.
• 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>Part</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>What</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>PDF</td>
<td><img src="https://www.snap4city.org/577" alt="PDF Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="PDF Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="PDF Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="PDF Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="PDF Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="PDF Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="PDF Image" /></td>
</tr>
<tr>
<td>Interactive</td>
<td><img src="https://www.snap4city.org/577" alt="Interactive Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Interactive Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Interactive Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Interactive Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Interactive Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Interactive Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Interactive Image" /></td>
</tr>
<tr>
<td>Video1</td>
<td><img src="https://www.snap4city.org/577" alt="Video1 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video2 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video3 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video4 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video5 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video6 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video7 Image" /></td>
</tr>
<tr>
<td>Video2</td>
<td><img src="https://www.snap4city.org/577" alt="Video2 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video3 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video4 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video5 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video6 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video7 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video7 Image" /></td>
</tr>
<tr>
<td>Video3</td>
<td><img src="https://www.snap4city.org/577" alt="Video2 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video3 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video4 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video5 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video6 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video7 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video7 Image" /></td>
</tr>
<tr>
<td>Video4</td>
<td><img src="https://www.snap4city.org/577" alt="Video2 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video3 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video4 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video5 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video6 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video7 Image" /></td>
<td><img src="https://www.snap4city.org/577" alt="Video7 Image" /></td>
</tr>
<tr>
<td>Duration</td>
<td>2:55</td>
<td>3:16</td>
<td>3:41</td>
<td>2:00</td>
<td>2:48</td>
<td>2:35</td>
<td>1:47</td>
</tr>
</tbody>
</table>
General Overview of the full Course 2021

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

A number of the training sections include exercitations

Updated versions on: https://www.snap4city.org/577
See also courses in ITALIANO: https://www.snap4city.org/485
**Agenda**

- **Solution Analysis and Design**
- **Data Model Capabilities and Architecture**
  - Data modeling, data type dictionary
- **Data Ingestion Strategy and Orientation**
- **Automated Data Ingestion**
  - Loading POI, Data Table Loader, Verification of Data Ingestion,
- **Verification of Data Ingestion**
  - Usage of: ServiceMap, Data Inspector, Digital Twin, DevDash My Data Dashboard Dev Kibana,
- **Technical Ingestion of Structural Data**
  - Set up of the road map
  - Loading POI via IoT Apps
  - Loading triples on Knowledge Base
- **Technical Data Ingestion Tools**
  - Data Ingestion architecture and Processes, IOT Apps
  - Data ingestion of static data: POI and GIS, KB
- **Data Ingestion of Time Series Devices via IOT Brokers**
  - IOT Directory, IOT Broker Registration, IOT Device Model, IOT Device Registration
- **Real Time Data Ingestion via IOT Applications**
  - Real Time Data Ingestion for Industry 4.0 Cases
  - Data Ingestion via API: External Services, using HTTP MicroService on IOT Applications
  - Data Ingestion via IOT Applications towards MyKPI, Web Scraping

- **Data Streams from Smart City API, participatory**
- **Data Streams from Mobile Devices**
- **Data Streams from Dashboards**
- **GIS Data Import and Export**
- **Integration with CKAN, open data manager and portal**
- **Integration with Copernicus Satellite Data**
- **Social Media data collection and exploitation**
- **Data Ingestion and Transformation via ETL processes (only for former versions of Snap4City)**
- **Acknowledgements**
Challenges: Requests and Deductions

Public Admin.
Pub. Admin: detection of critical conditions, improving services

Mobility Operators
Tune the service, reselling data and services, prediction

Commercial: customers prediction and profiles, promotions via ads

Tourism Museums
Tune the service, prediction

Smart City Engine
Services & Suggestions
Transport, Mobility, Commercial (retail), Tourism, Cultural

User profiling
Collective profiles, User segmentation

User Behavior
Crowd Sources

Data: Public and Private, Static and Real Time

Private: user movements, social media, crowd sources, commercial (retail)
Public: infomobility, traffic flow, TV cameras, flows, ambient, weather, statistic, accesses to LTZ, services, museums, point of interests, ...

API for SME

Personal Time Assistant
dynamic ticketing, whispers to save time and money, geoloc information, offers, etc.
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

Connect IOT/IOE
Upload context Open Data
Connect external Services
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
Monitor City Platform
Promote Applications & Dashboards
Manage Apps & Dashboards, User Engagement

Community Building

City Operators
City Operators
Resource Operators
Inhouse companies
Networking
Tutorials
documentation
Tech providers
workshops
Category Associations
Corporations
Advertisers
City Users
personal services
subscription to applications
collaborations
agreements
Networking

Collaborative Platform

Early Adopters
Start-ups
partnerships
Case Studies
Research groups
help desk
experiments
networking
tutorials
documentation
Tech providers
workshops
Category Associations
Corporations
Advertisers
City Users
personal services
subscription to applications
collaborations
agreements
Networking

Collaborative Platform

Early Adopters
Start-ups
partnerships
Case Studies
Research groups
help desk
experiments
networking
tutorials
documentation
Tech providers
workshops
Category Associations
Corporations
Advertisers
City Users
personal services
subscription to applications
collaborations
agreements
Networking

Collaborative Platform

Early Adopters
Start-ups
partnerships
Case Studies
Research groups
help desk
experiments
networking
tutorials
documentation
Tech providers
workshops
Category Associations
Corporations
Advertisers
City Users
personal services
subscription to applications
collaborations
agreements
Networking

Collaborative Platform

Early Adopters
Start-ups
partnerships
Case Studies
Research groups
help desk
experiments
networking
tutorials
documentation
Tech providers
workshops
Category Associations
Corporations
Advertisers
City Users
personal services
subscription to applications
collaborations
agreements
Networking
Solution Analysis and Design
Snap4City Innovation Matrix and Process

See Course Part 6
Data vs Smart Services enabling on Snap4City

- Public Transportation and mobility activated services in some where with Snap4City
  - Smart parking (parking locations, real time parking data, traffic, meteo) → predictions
  - Smart Fuel pricing (fuel station locations and real time prices)
  - Routing (detailed GIS information, text indexing of streets, POI, etc.)
    - Quite routing, perfect shopping, etc. etc. (more data in needed….)
  - Multimodal routing (detailed GIS information, Public transport time schedule)
  - Info traffic (traffic flow sensors, real time Traffic events, their localization, etc.)
  - Dense info traffic (traffic flow sensors and traffic flow reconstruction algorithm)
  - Car/Bike/Scooter Sharing (position and availability of Cars/Bikes, Scooters) … predictions
  - Smart Biking (cycling paths, environmental data) → predictions
  - E-vehicles (position, status of recharging stations, …) … predictions vs booking
  - Smart river crossing (position and status of Underpass, Ferry) … prediction
  - Quality of Public Transport (actual time of arrival at the bus stops, wrt planned time schedule)
  - Early Warning vs Resilience (combination of several data including mobility, events, Social to perform early warning…)
Data vs Smart Services enabling on Snap4City

- **Social and Users Behaviour**
  - Smart First Aid
  - search for POI and public transport services
  - Social Media Monitoring and acting
  - Information to Tourists
  - Early Warning, prediction of audience
  - Improvement of services for Tourists

- **Weather and environment, quality of life**
  - Weather forecast/condition
  - Air quality Pollution
  - Pollination
  - Alerting on Air quality for multiple parameters
  - Information Heatmaps for weather and air quality
  - Air quality indexes, and forecast

*Snap4City (C), January 2022*
Data Model Capabilities and Architecture
Development Life Cycle
Smart City Services

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

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

Deployment
- Deploy
- Testing
- Publication Production
URBAN PLATFORM: SMART CITY IOT AS A SERVICE AND ON PREMISE
Data Type Coverage

- POI, IOT, shapes,..
- maps, orthomaps, GTFS, GIS
- WFS/WMS, GeoTiff, ..
- calibrated heatmaps, ..
- traffic flow, typical trends, ..
- trajectories, events, ..
- 3D, BIM, Workflow, ..
- Dynamic icons/pins, ..
- OD Matrices, scenarios, ..
- prediction models, ..
- decision support, ..
- Synoptics, animations, ..
- social media, Routing, ..
- Satellite data, ..
- KPI, personal KPI,..
- etc.
Snap4City vs Formats

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

• Search the format you need to cope on the search box of Snap4City portal!: **Snap4City Supported Protocols, adding new protocols**
• **Open Data:**
  - Data gate, federation of Open Data Portals
  - IOT App, ETL proc (PULL)

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

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

• **Satellite data**

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

• **Mobile Apps**
  - Smart City API

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

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

• **Data base accesses**
  - GIS: WFS, WMS
  - ETL, IOT Application
Standards and Interoperability (2022)


https://www.snap4city.org/65
Data Inspector (Digital Twin info)

• Digital Twin (local/global)
  – 3D shapes
  – Device and sensors data
  – Values
  – Healthiness criteria and values
    • Machine learning tools
  – Images and physical world
  – Licensing
  – Users

• Users
  – Defined the Data and Devices
  – Defined the processes, IoT App
  – Created the dashboards
  – Etc.

• Process Views
  – Device Management tool
  – Data ingestion processes
    • ETL, IoT Apps
  – Data storage access views
    • Index views
    • Relationships view
  – Data Analytics and Transformation
    • IoT App, R Studio, Python
  – Data Rendering Dashboards
    • Synoptics
  – Processes’ Developers
The usage of IOT Applications

• IOT Applications = Node-RED + Snap4City Libraries

• Used for:
  – Data Ingestion, Transformation, Extract, Load, and Adaptation (format and protocol), See Part 5 of the Course (this part)
  – IOT Edge Devices logic, for implementing logic on IOT Edge, including IOT Device control (see on Part 3 of the course)
  – Business Logic control of Dashboards, via Web Sockets secure • see Part 2 of the Course
  – Control and schedule of Data Analytic, and Machine Learning (see part 4 of the Course)
  – Firing and condition identification and alerting.
Modalities and Strategies for data ingestion

- **Road Graphs**: from GIS, and/or OSM (see Snap4City tool for that), \( \rightarrow \) Km4City KB
- **Data** of any format via any protocol
- **Structured and non-structured** data (tables and free text, mixt)
- **Static data**/metadata descriptors ingested as: a file or manually via user interface:
  - POIs which have to be transformed in POIs by the Administrator
  - Open Data: Data Gate, a module of CKAN, allows to process files as POI, Sensors, etc.
  - IoT Devices to be registered on:
    - IOT Directory from user interface, ingesting them as Sensors
    - IOT Directory from file, ingesting them as a set of Sensors
- **Simple data and complex data** as: GIS data, heatmaps, tracks, etc.
- **Real Time Event Driven data** can be ingested by:
  - Automatically ingested from IoT Devices, via IoT Brokers
  - Dashboards, user events, etc.
  - Producing
    - IOT App/ETL processes for data adaptation, ingestion, gathering, harvesting, etc.
    - WebScraping processes
Snap4City Services also on IOT Edge!!!
Snap4City: IOT Directory and data/device Discovery

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

(IOT) Discovery
- Knowledge Base
- Semantic Reasoners

IOT Directory

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

Snap4City (C), January 2022
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
- HOW TO: save / export a MyKPI data into a CSV file
- HOW TO: produce heatmaps, custom heatmaps on any data
Further readings

- HOW TO: add a device to the Snap4City Platform
- HOW TO: add data sources to the Snap4City Platform
- HOW TO: add IOT Device data source from external broker to the platform.
- TC9.13: How to upload a local file into your IOT Application
- TC9.16 Web Scraping to get data from web pages
- TC6.3. Creating ETL processes for automated data ingestion and data transformation
- TC9.2. Managing heterogeneous File Ingestion, protocols, formats via IOT applications, and open standards
- TC6.1. Managing DataSets via DataGate: ingest, search, download, upload, annotate, share
- TC6.2. Search on DataGate for Data Sets
- TC2.25. Registering external MicroService calling RestCall services, using it on IOT applications

In Yellow alternative & legacy solutions

Snap4City (C), January 2022
Data Gathering and Knowledge Management

- Data ingestion can be performed by using multiple tools:
  - We suggest:
    - IOT App for static and real time data and flow, from IOT Brokers/Devices
    - DataGate for Static Data, upload them as files, or collected from other CKAN
    - WebScraper for scraping data from Web Pages, when authorized!
    - ETL for static / periodic data in PULL

- See how to test cases:
  - HOW TO: add data sources to the Snap4City Platform
  - HOW TO: define privacy rules for personal data, produced by the end-users own device
  - US6. Developing and using processes for data transformation
  - TC6.1 - Managing DataSets via DataGate: ingest, search, download, upload, annotate, share
  - TC6.3 - Creating ETL processes for automated data ingestion and data transformation
  - TC6.5 - Managing Heterogeneous File Ingestion via ETL processes
  - TC6.9 - ETL processes for multiprotocol and format data ingestion, see on GITHUB for library
  - TC9.2 - Managing heterogeneous File Ingestion, protocols, formats via IOT applications, and open standards

In Yellow alternative & legacy solutions
Data Concepts, HLT
High Level Types
What happens to data into the platforms

• **Static** information should be collected in advance
  – The maps, the kpi, the past historical data can be also static
  – collection of IoT devices (device), data sources, etc.
  – Supporting and facilitating the search of relationships among entities

• **Dynamic** information to be connected to Static one at the ingestion time
  – Data updates, measured values, etc.

• **PLUS**: when data enter into the platform a wide number of information is added/created such as:
  – Internal technical identification, modalities to access to the information
  – Historical data storage
  – Eventual derived data: date and time of ingestion, average, quality level, etc.
  – Rights to access at the data: who can access and to do what?
  – Visual representation of data, may be an image of the Device how it has been installed and where, by who, telephone number for maintenance, etc.
  – Process ID used to ingest the data, possible IOT Applications that are using the data, ...
Static vs Dynamic Information

• Most of the valuable data are **Real Time/Dynamic data**, based on **Static** info.

• **For example**, data coming from Smart Park or a Smart Light solution
  – They are composed of:
    • **One or more IoT Device models, types, kinds**..... common information such as: licensing, gateway access, gateway credentials, description of provider, time of update, etc.
      – The set/collections of Devices is produced from the model as a template
    • **each** IoT **Device** has **Static** information such as:
      – data structure: device ID, Current consumption, temperature, ......
      – GPS coordinates... (if the devices is not a moving one)
      – Classification: nature, subnature, Healthiness criteria, MTTF, etc.
    • **Each Device** produces a data package/message according to minimum data Structure as **Dynamic** information, in PUSH towards the Gateway, including real time data
  – The Gateway in turn can be:
    • a sort of IoT Broker sending data to other consumer in Push or
    • can provide also data in PULL when requested by some consumer via some API, for example.

• This means that data fully describing the situation is composed by **Static** and **Dynamic** aspects that have to described into the platform and reconnected when one access to the device...
Data are not so Simple as one can imagine

• Data to be managed into the Smart City IOT are not so simple as one may imagine, and not limited to take into account only IOT Devices.
• THUS, a large number of data TYPES and sources have also to be addressed:
  – E.g.: external services, heatmaps, trajectories, maps, OD matrices, actuators, personal data, KPI, API descriptors, special widgets, events, predictions, Tweets, posts, GIS, mobile devices, etc.
  – With their complexity of managing data, licensing, etc...
• THEY are called High Level Types: how and which tool / process can cope with them into the Smart City Platform?... See next!
  – Thus a unified model is needed.
• In 2nd part we have shown how to show specific types of data
**Data Type Coverage**

- POI, IOT, shapes, ..
- maps, orthomaps, GTFS, GIS WFS/WMS, GeoTiff, ..
- calibrated heatmaps, ..
- traffic flow, typical trends, ..
- trajectories, events, ..
- 3D, BIM, Workflow, ..
- Dynamic icons/pins, ..
- OD Matrices, scenarios, ..
- prediction models, ....
- decision support, ....
- Synoptics, animations, ..
- social media, Routing, ..
- Satellite data, ..
- KPI, personal KPI,..
- etc.
The main High Level Types, HLT, Entities

- **IoT Device Mobile**: dynamic GPS, Info, variable data, Time Series
  - Sensors and actuators
  - **Identify By**: Model, Device Name, (Classification), ValueName (Variable)
- **IoT Device**: static GPS, Info, variable data, Time Series
  - Sensors and actuators
  - **Identify By**: Model, Device Name, (Classification), ValueName (Variable)
- **Data Table**: IoT Devices created from Excel Files, Time Series (it could be a way to start loading IoT Devs.)
  - Sensors and actuators
  - **Identify By**: Model, Device Name, (Classification), ValueName (Variable)
- **Sensors**: data, a sort of IoT Devices, may be taken from ETL, with Time Series
  - Sensors
  - **Identify By**: Device Name, (Classification), ValueName (Variable, Sensor)
- **MyKPI**: dynamic GPS, info, single variable, Time Series, (Classification),
- **POI**: static GPS, info about a location, (Classification),
- **Heatmaps**: matrices on some area, Time Series, (Classification),
- **Traffic Flow**: road segments with flow density, Time Series
- **OD Matrices**: ...
- **BIM representations**: ...

Snap4City (C), January 2022
Any Entity has a Semantic Classification

Nature
- Accommodation
- Advertising
- AgricultureAndLivestock
- CivilAndEdilEngineering
- CulturalActivity
- EducationAndResearch
- Emergency
- Entertainment
- Environment
- FinancialService
- GovernmentOffice
- HealthCare
- IndustryAndManufacturing
- IoTDevice
- MiningAndQuarrying
- ShoppingAndService
- TourismService
- TransferServiceAndRenting
- UtilitiesAndSupply
- Wholesale
- WineAndFood

SubNature
- EducationAndResearch
- Educational_support_activities
- Higher_education
- Languages_courses
- Performing_arts_schools
- Post_secondary_education
- Pre_primary_education
- Primary_education
- Private_high_school
- Private_infant_school
- Private_secondary_school

Snap4City (C), January 2022
• Conceptually are IoT Devices with sensors/actuators, IN/IN-OUT

• They are classified in terms of nature/subnature

• For Searching and showing on maps and dashboards

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

IoT Device Variables
  • dateObserved: ............
  • ID:
  • Status: ready
  • Temperature: 70%
  • WaterLevel: 35%
  • UsedCapsBox: 30%
  • Power: OK
  • .......
What About IoT Devices, Time Series

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

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

Each new measure in Snap4City is conventionally time located in «dateObserved», which has to be Unique.
- Only one message per dateObserved is allowed

<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>
Time Series: they are data streams

• As soon as you have registered an IoT Device
  – You are ready to get Future data, may be arriving in PUSH
  – Recall and store historical data as well, but they have to be
    • recalled in PULL with some IoT App.
    • Loaded in PULL with some File or Data Table Loader
Mobile Devices

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

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

**Mobile Device Models**

**Mobile Device**
- Name:......
- Model:......
- Spec:....

**Mobile Device Variables**
- ID:
- `dateObserved`: ........
- Status: ready
- Temperature: 70%
- Gasoline: 35%
- Velocity: 231.3 Km/h
- Position: 44.3223, 11.3432
- .....
• When you have many **IoT Devices or Virtual Devices**, you may have them listed with their information in some data table
  – Then you can load them in short time via **Data Table Loader** tool, to produce:
    • Data Table Model, Data Table Device, Data Table Variable
    • with the same corresponding meanings of **IoT Devices and Mobile Devices**.

• **Data Tables** are a just **a special case of IoT Devices**, which have not been created manually or via some broker but at the end are
  – managed as IoT Devices, Mobile Devices in the system
  – Once created from the Data Table Loader,
  – they can be received from some IoT Orion Broker

• They are classified in terms of nature/subnature
• For Searching and showing on maps and dashboards, they are identical to **IoT/Mobile Devices** can be:
  – **Data Table Model**, for example: «sedan», «personal coffeeemachine»
  – **Data Table Device** name, for example: «BMW JD7356HD», «Ford KO786KK»
  – **Data table Variable**, for example: «velocity», «temperature»
Sensor/Sensor-Actuator

Sensor Device
- Name:......
- Model:......
- Position: ......

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

- They are classified in terms of nature/subnature
- For Searching and showing on maps and dashboards

HLT of Sensors/Sensor-Actuator can be:
- **Sensor Device** name, for example: «mycoffemachine1», «CM23»
- **Sensor/sensor-actuator** is a variable of a Sensor Device, for example: «Temperature»

- They do not have a model, while, in KB, have a reference process from which their real time data are collected from the field, from gateways, etc.
POI, Point of Interest

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

- Do not have Time Series for variable over time
- May sporadically change over time
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
**Data Ingestion Strategy**

- **Structural Data**: Maps, orthomaps, geolocations, roads, etc.
  - Typically arrive as database, GIS data, etc.
  - **Suggested approaches**: IoT App, OSM → SM, ETL

- **POI (point of interest)**: info point with geolocation as services, museums, restaurants, banks, email, urls, etc.
  - Typically arrive as: excel files, GIS data, etc.
  - **Suggested approaches**: POI Loader, IoT App, ETL

- **IoT Devices, Data Tables, ... (Devices and Virtual Devices/KPI)**, including
  - Description, including geolocations, etc.
  - Time Series: measures that change over time,
  - They can also move → **IoT Device Mobile, Data Tables**
  - Typically arrive as:
    - description and real time values or additional values
    - Excel files with description and data all together
  - **Suggested approach**: Data Table Loader, IoT App, Brokers, ETL
    - IoT Brokers also send data in real time

**ETL**: deprecated approach in V2
KPIs, Key Performance Indicators

• They are classified in terms of nature/subnature
• Typically associated with
  – City or infrastructure, so that the GPS can be city center
  – Some date: 2019, March 2019, etc.
• For example:
  – Number of Arrivals from France in March 2019
  – Average price for **** hotels in 2019, downtown
  – Net income of the region
  – CO2 saved in the April 2020
  – Total number of vehicles sold in 2020
  – Stock option value of Airport
• Note that in most cases:
  – They can be managed as Virtual IoT Devices
<table>
<thead>
<tr>
<th><strong>Single Variable for MyKPI</strong></th>
<th><strong>an IoT Device may have multiple Sensors/variables</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HLT: MyKPI</strong></td>
<td><strong>HLT: IoT Device, Sensor Device, Data Tab Device</strong></td>
</tr>
<tr>
<td><strong>Nature:</strong> Industry and manufacturing</td>
<td><strong>Nature:</strong> Industry and manufacturing</td>
</tr>
<tr>
<td><strong>Subnature:</strong> Chemical</td>
<td><strong>Subnature:</strong> Chemical</td>
</tr>
<tr>
<td><strong>Value Name:</strong> CloroParaffine</td>
<td><strong>Value Name:</strong> Irrigator fioriera Gag</td>
</tr>
<tr>
<td><strong>Value Type:</strong> Density percentage</td>
<td><strong>Value Type:</strong> Battery Level</td>
</tr>
<tr>
<td><strong>Value Unit:</strong> %</td>
<td><strong>Value Unit:</strong> V</td>
</tr>
<tr>
<td><strong>Data Type:</strong> float mykpi</td>
<td><strong>Data Type:</strong> float</td>
</tr>
<tr>
<td><strong>Last Date:</strong> 2019-02-25</td>
<td><strong>Last Date:</strong> 2020-04-01 12:59:00</td>
</tr>
<tr>
<td><strong>Last Value:</strong> 87.0</td>
<td><strong>Last Value:</strong> 5.18</td>
</tr>
<tr>
<td><strong>Healthiness:</strong></td>
<td><strong>Healthiness:</strong></td>
</tr>
<tr>
<td><strong>Last Check:</strong> 2020-04-03 10:28:12</td>
<td><strong>Last Check:</strong> 2020-04-03 03:28:12</td>
</tr>
<tr>
<td><strong>Ownership:</strong> private for xyz...</td>
<td><strong>Ownership:</strong> public/private</td>
</tr>
<tr>
<td><strong>Organization:</strong> Firenze</td>
<td><strong>Organization:</strong> Firenze</td>
</tr>
</tbody>
</table>

This Section is repeated for each variable
**HLT: Unified Classification for Data and Services**

- **Data Models**: all devices sprunt from that model
  - IoT Device Model, Mobile Device Model, Data Table Model
- **Devices**: are instances of some model or sprunt from processes
  - IoT Device, Mobile Device, Data Table Device, Sensor Device
- **Variables**, **Sensor/sensor-actuator**:
  - IoT Device Variable, Mobile Device Variable, Data Table Variable, Sensor, Sensor-Actuator
  - **Dashboard-IOT App**: messages from GUI to Business Logic on IoT App
- **MyKPI**: dynamic GPS, info, single variable, Time Series, *(Classification)*
  - **KPI**: former KPI model
  - **MyPersonaData/MyData**: safes in which specific personal data are saved.
- **POI**: static GPS, info about a location, *(Classification)*,
  - **MyPOI**: personal POI that can be leveraged to standard POI by administrator
- **Heatmaps**: matrices on some area, Time Series, *(Classification)*
- **Traffic Flow**: road segments with flow density, Time Series, *(Classification)*
- **OD Matrices**: origin destination matrices, Time Series, *(Classification)*
- **Complex events**: emergency, alarm, entertainment, CAP, ... special widgets
**HLT: Unified Classification for Data and Services**

- **External Service**: third party visualization tools, iFRAMED...
  - Also TV CAMs are rendered here, and substantially all the other Services
- **Synoptics**: graphic representations with animation connected to variables and/or MyKPI and/or IoTApp, etc.
- **BIM representations**: Digital Twin Local, ...
- **Micro Applications**: Snap4City, Km4City micro applications, iFRAMED
- **Special Widget**: a set of special visualization tool with their dedicated data type
- **WFS**: a specific tool for WFS GIS rendering, please note almost the same kind of data type can be visualized as Data above described

Snap4City (C), January 2022
Data Inspector: HLT classification
## HLT: Unified Classification for Data and Services

<table>
<thead>
<tr>
<th>High Level Types</th>
<th>Nature</th>
<th>SubNature</th>
<th>Technical Source</th>
<th>Variables, names</th>
<th>Value Name</th>
<th>Value Type</th>
<th>Data Type</th>
<th>Value Unit</th>
<th>Last Date/Time</th>
<th>Last Value</th>
<th>Healthiness</th>
<th>Last Check</th>
<th>Ownership</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table:**

<table>
<thead>
<tr>
<th>Device/Model</th>
<th>Broker</th>
<th>Value Name</th>
<th>Value Type</th>
<th>Data Type</th>
<th>Value Unit</th>
<th>Last Check</th>
<th>Healthiness</th>
<th>Last Date/Time</th>
<th>Last Value</th>
<th>Ownership</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Snap4City (C), January 2022
### How to Ingest

All of them can be shown on Dashboards, what about manipulate them!!!!

<table>
<thead>
<tr>
<th>HLT, High Level Types++</th>
<th>GPS</th>
<th>Static</th>
<th>Dynamic</th>
<th>MacroCat</th>
<th>Single</th>
<th>Time Series</th>
<th>Trajectory</th>
<th>HTTP</th>
<th>How to ingest/change/manage</th>
</tr>
</thead>
<tbody>
<tr>
<td>IoT Device Model</td>
<td>--</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Broker, IOT App, Data Table, API, UserInterf,</td>
</tr>
<tr>
<td>Mobile Device Model</td>
<td>--</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Broker, IOT App, Data Table, API, UserInterf,</td>
</tr>
<tr>
<td>Data Table Model</td>
<td>--</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Data Table, then as IoT Device</td>
</tr>
<tr>
<td>IoT Device</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>Broker, IOT App, Data Table, API, UserInterf,</td>
</tr>
<tr>
<td>IoT Device Variable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Into the corresponding device</td>
</tr>
<tr>
<td>Mobile Device</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>Broker, IOT App, Data Table, API, UserInterf,</td>
</tr>
<tr>
<td>Mobile Device variable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Into the corresponding device</td>
</tr>
<tr>
<td>Data Table Device</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>Data Table, then as IoT Device</td>
</tr>
<tr>
<td>Data Table device Variable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Into the corresponding device</td>
</tr>
<tr>
<td>Sensor Device</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>Dashboard, IOT App, UserInterf, API, ...</td>
</tr>
<tr>
<td>Sensor / Sens. Actuator</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Into the corresponding device</td>
</tr>
<tr>
<td>MyKPI</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>Dashboard, IOT App, UserInterf, API, ...</td>
</tr>
<tr>
<td>KPI (metrics) data</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Dashboard, IOT App, API, Metrics SQL calls</td>
</tr>
<tr>
<td>My Personal Data</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Dashboard, IOT App, UserInterf, API, ...</td>
</tr>
<tr>
<td>POI (Point of Interest)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DataGate, ETL, IOT App, API, ...</td>
</tr>
<tr>
<td>MyPOI data</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Dashboard, IOT App, UserInterf, API, ...</td>
</tr>
<tr>
<td>Dashboard-IOT App (msg)</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Dashboard, IOT App, API, ...</td>
</tr>
<tr>
<td>Complex Event (msg)</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Dashboard, ETL, Special, IOT App, ...</td>
</tr>
<tr>
<td>HLT, High Level Types++</td>
<td>GPS</td>
<td>Static</td>
<td>Dynamic</td>
<td>MacroCat</td>
<td>Single</td>
<td>Time Series</td>
<td>Trajectory</td>
<td>HTTP</td>
<td>How to ingest/change/manage</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----</td>
<td>--------</td>
<td>---------</td>
<td>----------</td>
<td>--------</td>
<td>-------------</td>
<td>-----------</td>
<td>------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>Heatmap matrix</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>IOT App, MicroService, UserInterf, API, …</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Traffic Flow</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>IOT App, MicroService, UserInterf, API, …</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OD Matrix</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>IOT App, MicroService, UserInterf, API, …</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| External Service (web pag) | Yes | --    |         |         |         |             | Yes       | ETL, Special, IOT App, Web Scraper, … |
| Synoptics (groups)        | Yes | (Yes) | (Yes)   | (Yes)   | (Yes)   |             | Yes       | Special, API, … |
| BIM View                 | Yes | Yes   | (Yes)   | (Yes)   | (Yes)   |             | Yes       | Special, API, … |
| BIM Device               | Yes | Yes   | Yes     | Yes     | Yes     |             | Yes       | Special, API, … |
| MicroApplication (webapp) | Yes | --    |         |         |         |             | Yes       | Dashboard, IOT App, API, FTP, … |
| Special Widget (complex) | Yes | Yes   | Yes     | Yes     | Yes     |             | Yes       | ETL, special, IOT App, API, … |
| WFS/WMS (GIS data)       | Yes | [yes] | [yes]   | Yes     | Yes     |             | Yes       | GIS tools, or GeoServer, IOT App, … |

| Tools (functional)        | (Yes) | (Yes) | (Yes)   | (Yes)   | MDM     | Scenarios, What-If, etc. |

All of them can be shown on Dashboards, what about manipulate them!!!!
Dictionary for Data Fields
Semantics and Technical Meaning
Unified Data and Services Model/Classification

**Semantic Nature**

- SubNature
- SubNature

**Technical meaning**

- Value Unit
- Value Unit

- Exists a Dictionary for the 4 categories
- They are related each other and not all values are possible
- Right setting lead to right rendering on graphs and automated combinations and processing
- The Dictionary is used by many tools

Snap4City (C), January 2022
For example

Technical meaning

Value Type

Power

Value Unit
mW

Value Unit
KW

Data Type
Integer

Data Type
Float

Link to Friend Sensor as ServiceURI:

Value Type

Value Unit
URL

Value Unit
KW

Data Type
String, URL

Data Type
Float
Example of Energy and its Value Units

Value Type: Energy

- Watt per hour
- KiloWatt per hour
- MegaWatt per hour
Please note on: Data Type

• Value Types have only a few number of Data Types because they represent how the data area treated into the system

• Therefore main Data Types are:
  – **Float**: numbers with decimals large as you like, etc.
  – **Integer**: numbers, booleans (0/1), etc.
  – **String**: url, links, names, id, descriptions, status code, etc.
  – **Json**: structured data, vector, matrices, etc.

  – …
Data Ingestion Strategy and Orientation
The main High Level Types

- **IoT Device Mobile**: dynamic GPS, Info, variable data, Time Series
  - Sensors and actuators
- **IoT Device**: static GPS, Info, variable data, Time Series
  - Sensors and actuators
- **MyKPI**: dynamic GPS, - info, single variable, Time Series
- **POI**: codified metadata, static GPS, + info, no time series
- **Heatmaps**: matrices on some area, Time Series
- **Traffic Flow**: road segments with flow density, Time Series
- ...
# Data Ingestion Strategy

<table>
<thead>
<tr>
<th>HLT</th>
<th>GPS &amp; Geo References</th>
<th>Info Meta Data</th>
<th>Variables Value Name</th>
<th>Time Serie, Seq.</th>
<th>In/Out Read/Write</th>
<th>Manual Ingestion Tool</th>
<th>Automated Loading tools</th>
<th>Technical IoT App management</th>
</tr>
</thead>
<tbody>
<tr>
<td>IoT Device Mobile</td>
<td>Yes Moving</td>
<td>yes</td>
<td>Multiple</td>
<td>yes</td>
<td>yes</td>
<td>Yes (IoT Directory)</td>
<td>IoT Brokers</td>
<td>Yes (IoT App..)</td>
</tr>
<tr>
<td>IoT Device</td>
<td>Yes only once</td>
<td>yes</td>
<td>Multiple</td>
<td>yes</td>
<td>yes</td>
<td>Yes (IoT Directory)</td>
<td>Yes (Data Table Loader)</td>
<td>Yes (IoT App..)</td>
</tr>
<tr>
<td>MyKPI</td>
<td>Yes Moving</td>
<td>yes</td>
<td>1</td>
<td>yes</td>
<td>yes</td>
<td>Yes (MyKPI editor)</td>
<td></td>
<td>Yes (IoT App..)</td>
</tr>
<tr>
<td>POI</td>
<td>Yes</td>
<td>yes</td>
<td>No, only static</td>
<td>No</td>
<td>Write 1, read</td>
<td>No / Yes via MicroApplication</td>
<td>Yes (POI Loader)</td>
<td>Yes (IoT App..), CKAN, etc.</td>
</tr>
<tr>
<td>Heatmaps</td>
<td>Area, matrix</td>
<td>yes</td>
<td>1 on the matrix</td>
<td>yes</td>
<td>Write 1, read</td>
<td>Load on Heatmap Manager</td>
<td>Yes via Rstudio, Python, IoT App</td>
<td>Yes (IoT App..)</td>
</tr>
<tr>
<td>Traffic Flow</td>
<td>roads</td>
<td>yes</td>
<td>1-4: traffic flow density, velocity,...</td>
<td>yes</td>
<td>Write 1, read</td>
<td>Load on Traffic Flow Manager</td>
<td>Special tools</td>
<td>Yes (IoT App..)</td>
</tr>
<tr>
<td>OD Matrix</td>
<td>Flow, mat</td>
<td>yes</td>
<td>1 on the matrix</td>
<td>Yes</td>
<td>Write 1, read</td>
<td>Load on OD Manager</td>
<td>Special tools, IoT App</td>
<td>Yes (IoT App..)</td>
</tr>
</tbody>
</table>
How they can be ingested

• Main Concepts
  – If it is NOT A TIME SERIES
    • We have only to perform the static/quasi static registration
    • POI \(\rightarrow\) POI Loader, also a MyKPI, or a MyPOI, or an IoT Device...
  – If it is Time Series: (IoT Device, IoT Device Mobile, MyKPI, Heatmaps, etc.)
    • the static information has to be registered ONCE
    • At each time instant (dateObserved) time varying data is stored
    • In the next slides the difference cases are presented
    • Multiple Solutions are needed for their ingestion
Data Ingestion via Excel Files

• **TWO main factors:**
  
  – **dateObserved (Yes/no)** is the date/time at which the data refers / are measured (not the date of loading which is always automatically set):
    
    • YES → they may be the
      
      – **Single:** Year (2019), or month (March 2019), or full time stamp in ISO standard with year, month, date: HH:MM:SS, ms
      
      – **Time series:** one or many samples along time, for the same device, for the same structure
    
    • NO → they are static information, may be a POI
  
  – **GPS (yes or no), (lat, lon)**
    
    • YES → data is geolocalized or refer to a certain point of the city,
      
      – thus GPS coordinates of the point of the city/area center can be taken
    
    • NO
      
      – A) May be assigned to the center of the city if it is a KPI or an Origin Destination matrix (by grid, by shapes, by administrative structured/shapes)
      
      – B) If not possible to map it is better to manage them as MyKPI
Main IoT Data In/Out flows

IOT Orion Broker

Knowledge Base
Semantic Reasoners

Indexing and Aggregating
NIFI, OpenDistro per ElasticSearch

IOT Device Model Reg.

IOT Device Registration

IOT Directory

registration triples

NGSI

subscription note

IOT Orion Broker

NGSI

Real Time

SURI Link

IOT App

IOT App

IOT App

Data Manager

MyKPI

IOT App

NGSI
Automated Data Ingestion
## Data Ingestion Strategy

<table>
<thead>
<tr>
<th>HLT</th>
<th>GPS &amp; Geo References</th>
<th>Info</th>
<th>Variables Value Name</th>
<th>Time Serie, Seq.</th>
<th>In/Out Read/Write</th>
<th>Manual Ingestion Tool</th>
<th>Automated Loading tools</th>
<th>Technical IoT App management</th>
</tr>
</thead>
<tbody>
<tr>
<td>IoT Device Mobile</td>
<td>Yes Moving</td>
<td>yes</td>
<td>Multiple</td>
<td>yes</td>
<td>yes</td>
<td>Yes (IoT Directory)</td>
<td>IoT Brokers</td>
<td>Yes (IoT App..)</td>
</tr>
<tr>
<td>IoT Device</td>
<td>Yes only once</td>
<td>yes</td>
<td>Multiple</td>
<td>yes</td>
<td>yes</td>
<td>Yes (IoT Directory)</td>
<td></td>
<td>Yes (IoT App..)</td>
</tr>
<tr>
<td>MyKPI</td>
<td>Yes Moving</td>
<td>yes</td>
<td>1</td>
<td>yes</td>
<td>yes</td>
<td>Yes (MyKPI editor)</td>
<td>Yes (Data Table Loader)</td>
<td></td>
</tr>
<tr>
<td>POI</td>
<td>Yes</td>
<td>yes</td>
<td>No, only static</td>
<td>No</td>
<td>Write 1, read</td>
<td>No / Yes via MicroApplication</td>
<td>Yes (POI Loader)</td>
<td>Yes (IoT App..), CKAN, etc.</td>
</tr>
<tr>
<td>Heatmaps</td>
<td>Area, matrix</td>
<td>yes</td>
<td>1 on the matrix</td>
<td>yes</td>
<td>Write 1, read</td>
<td>Load on Heatmap Manager</td>
<td>Yes via Rstudio, Python, IoT App</td>
<td></td>
</tr>
<tr>
<td>Traffic Flow</td>
<td>roads</td>
<td>yes</td>
<td>1-4: traffic flow density, velocity...</td>
<td>yes</td>
<td>Write 1, read</td>
<td>Load on Traffic Flow Manager</td>
<td>Special tools</td>
<td>Yes (IoT App..)</td>
</tr>
<tr>
<td>OD Matrix</td>
<td>Flow, mat</td>
<td>yes</td>
<td>1 on the matrix</td>
<td>Yes</td>
<td>Write 1, read</td>
<td>Load on OD Manager</td>
<td>Special tools, IoT App</td>
<td></td>
</tr>
</tbody>
</table>

Snap4City (C), January 2022
Short cut Data Ingestion from Excel file

Snap4City Architecture

Data Loader

- Data Table Loader
- IOT App for Data Table Loader

IOT Orion Broker

- IOT Orion Broker
- IOT Directory
- Dictionary

Knowledge Base

- Semantic Reasoners
- Indexing and Aggregating
  - NIFI, OpenDistro per ElasticSearch

Dashboard Builder

Real Time data

- NGSI
- IOT Device Time Series
- IOT Device Registration
- IOT Device Model Registration
- Triples registration

NGSI

- Subscription
- Real Time
- SURI Link
POI Loader, from Excel Files
(for authorized AreaManagers)

https://www.snap4city.org/731
POI, Point of Interest

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

- Do not have Time Series for variable over time

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

• To **enable you to**
  – create dashboards from them according to different views and nature

**POI Loader**

[Diagram showing data flow from Excel to IOT App to File with Triples to POI Table to KB]
POI Structure, EXCEL

- nameENG, abbreviationENG, descriptionShortENG, descriptionLongENG
- Phone, Fax,
- url (web page), email
- refPerson
- secondPhone, secondFax, secondEmail, secondCivicNumber, secondStreetAddress
- notes
- timetable
- photo
- Other1, other2, other3
- Postalcode, Province, city
- streetAddress, civicNumber
- Latitude, longitude


https://www.snap4city.org/drupal/system/files/private/POI_loader_template.zip

Snap4City (C), January 2022
Note on POI Loader

- The structure of Excel File is fixed
- UTF8 characters can be used into the values and not in the device name neither on field names
- Follow the guidelines in the first page and the instruction during the upload
- POI Loading is performed via an IoT App which produces triples
  - They are verified and loaded by some administrator
- Any AreaManager can upload POI data sets but only specific dedicated responsible for data upload can actually load being owners and responsible of the IOT App process, which can be customized also.
Data Table Loader: Upload IoT Device data via Excel Files
(for Authorized AreaManagers)
Data Table Loader

- For: IoT Devices, KPI as devices, OD as devices, time Series ...
- To help you to **upload data in short/zero time**
  - Start from Excel Files, they should be formatted some how or well formatted according to our guidelines (models are provided)
  - Custom upload for each Organization is possible
- To **enable you** to
  - create dashboards from them according to different views and nature
Short cut Data Ingestion from Excel file

1) Upload the file on Data Table Loader
2) Follows the instructions and guidelines – the dirty work will be done in a Snap – wait!
3) See data on your Data Inspector 😊
4) Use Data Into Dashboards 😊
Assuming **an Excel file with 1 or more Sheets**

**all of them with the same structure**

<table>
<thead>
<tr>
<th>Sheet1</th>
<th>Sheet2</th>
<th>Sheet3</th>
</tr>
</thead>
<tbody>
<tr>
<td>a b c d e f</td>
<td>a b c d e f</td>
<td>a b c d e f</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*If this is not the case!*  
- **Columns in the sheets are different AND it is not possible to regularize them** *(by editing: adding empty columns, change names, etc.)*  
  → split the excel file in multiple files
**Assuming an Excel file with 1 or more Sheets all of them with the same structure**

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**If this is the case!**

1) The Schema of the sheets is becoming an IoT Device Model
2) Each single Sheet is becoming a single IoT Device
3) The row in the single Sheet are becoming instances of the corresponding IoT Device
Row Case

• One dateObserved for each Row of each Sheet, multiple sheets with the same structure
  – An IOT Device Model is created for the structure of the sheet
    • a number of variables are produced
    • Including dateObserved variable which is a column
  – For each Sheet an IOT Device is produced from the model
    • The device name is the combination: sheet+............. as defined by the user
      – Sheet name is part of the IOT Device Name and may also become a variable
      – Sheets may have a different number of rows
    • For each Row of each Sheet an IOT Device Time Instance is created
      – Each row has a specific dateObserved
      – Each row has ... other attributes as well
# Basic Sizes of Incoming Tourism of the Region of Western Greece 2019

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>West Hellas</td>
<td>Albania</td>
<td>132.9</td>
<td>26.3</td>
<td>225.8</td>
<td>199.7</td>
<td>137.5</td>
<td>1.7</td>
</tr>
<tr>
<td></td>
<td>United Kingdom</td>
<td>47.7</td>
<td>17.9</td>
<td>345.8</td>
<td>375.2</td>
<td>51.8</td>
<td>7.2</td>
</tr>
<tr>
<td></td>
<td>Germany</td>
<td>70.3</td>
<td>36.4</td>
<td>672.4</td>
<td>517.9</td>
<td>54.1</td>
<td>9.6</td>
</tr>
<tr>
<td></td>
<td>France</td>
<td>55.4</td>
<td>16.5</td>
<td>321.6</td>
<td>298.1</td>
<td>51.4</td>
<td>5.8</td>
</tr>
<tr>
<td></td>
<td>Other</td>
<td>510.7</td>
<td>160.0</td>
<td>2,964.9</td>
<td>313.3</td>
<td>54.0</td>
<td>5.8</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>817.0</td>
<td>257.4</td>
<td>4,530.4</td>
<td>315.0</td>
<td>56.8</td>
<td>5.5</td>
</tr>
<tr>
<td>% of the total</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Source: BgG Border Research, INSETE Intelligence Editing

# Basic Sizes of Incoming Tourism of the Region of Western Greece 2018

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>West Greece</td>
<td>Albania</td>
<td>138.7</td>
<td>29.0</td>
<td>222.9</td>
<td>209.2</td>
<td>130.1</td>
<td>1.6</td>
</tr>
<tr>
<td></td>
<td>United Kingdom</td>
<td>42.6</td>
<td>13.5</td>
<td>180.6</td>
<td>317.6</td>
<td>74.9</td>
<td>4.2</td>
</tr>
<tr>
<td></td>
<td>Germany</td>
<td>71.3</td>
<td>26.0</td>
<td>466.5</td>
<td>365.1</td>
<td>55.8</td>
<td>6.5</td>
</tr>
<tr>
<td></td>
<td>France</td>
<td>44.2</td>
<td>13.3</td>
<td>282.9</td>
<td>304.7</td>
<td>51.2</td>
<td>6.0</td>
</tr>
<tr>
<td></td>
<td>Other</td>
<td>402.5</td>
<td>129.8</td>
<td>1,003.7</td>
<td>324.4</td>
<td>63.8</td>
<td>5.1</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>699.2</td>
<td>211.8</td>
<td>3,183.5</td>
<td>302.9</td>
<td>66.5</td>
<td>4.6</td>
</tr>
<tr>
<td>% of the total</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Resulted Data Table Loaded by Row Model

<table>
<thead>
<tr>
<th>Value Type</th>
<th>Count</th>
<th>price</th>
<th>Count</th>
<th>price</th>
<th>duration</th>
<th>timestamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>geoLocation</td>
<td>float</td>
<td>float</td>
<td>float</td>
<td>float</td>
<td>float</td>
<td>float</td>
</tr>
<tr>
<td>text</td>
<td>float</td>
<td>float</td>
<td>float</td>
<td>float</td>
<td>float</td>
<td>float</td>
</tr>
<tr>
<td>Snap4City (C), January 2022</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Data Table

<table>
<thead>
<tr>
<th>Device Name</th>
<th>Sheet Name</th>
<th>Region</th>
<th>Visitors_in_Thousands</th>
<th>Receipts_in_Millions_Euro</th>
<th>Nights_in_Thousands</th>
<th>Expenditure_per_Vist_Euro</th>
<th>Cost_per_Night_Euro</th>
<th>Average Length_of_Stay</th>
<th>dateObserved</th>
<th>Latitude</th>
<th>Longitude</th>
<th>Nature</th>
<th>Sub-Nature</th>
<th>Context Broker</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basics</td>
<td>것을 입력하세요</td>
<td>아테네</td>
<td>773.3</td>
<td>10.1</td>
<td>198.6</td>
<td>18.3</td>
<td>77.4</td>
<td>2.1</td>
<td>2016-12-31T00:00:00+02:00</td>
<td>35.3964</td>
<td>21.7985</td>
<td>Tourist</td>
<td>Information</td>
<td>null</td>
</tr>
<tr>
<td>Basics</td>
<td>것을 입력하세요</td>
<td>아테네</td>
<td>793.3</td>
<td>10.1</td>
<td>198.6</td>
<td>18.3</td>
<td>77.4</td>
<td>2.1</td>
<td>2017-12-31T00:00:00+02:00</td>
<td>35.3964</td>
<td>21.7985</td>
<td>Tourist</td>
<td>Information</td>
<td>null</td>
</tr>
<tr>
<td>Basics</td>
<td>것을 입력하세요</td>
<td>아테네</td>
<td>793.3</td>
<td>10.1</td>
<td>198.6</td>
<td>18.3</td>
<td>77.4</td>
<td>2.1</td>
<td>2018-12-31T00:00:00+02:00</td>
<td>35.3964</td>
<td>21.7985</td>
<td>Tourist</td>
<td>Information</td>
<td>null</td>
</tr>
<tr>
<td>Basics</td>
<td>것을 입력하세요</td>
<td>아테네</td>
<td>793.3</td>
<td>10.1</td>
<td>198.6</td>
<td>18.3</td>
<td>77.4</td>
<td>2.1</td>
<td>2019-12-31T00:00:00+02:00</td>
<td>35.3964</td>
<td>21.7985</td>
<td>Tourist</td>
<td>Information</td>
<td>null</td>
</tr>
</tbody>
</table>
Requested information if not provided

GeoLocation to assign at the city or area
• Latitude
• Longitude

Classification to simplify the search
• Nature
• Subnature

Broker is assigned automatically on the basis of Organization / tenant

<table>
<thead>
<tr>
<th>Latitude</th>
<th>Longitude</th>
<th>Nature</th>
<th>Sub-Nature</th>
<th>Context Broker</th>
</tr>
</thead>
<tbody>
<tr>
<td>38.2384</td>
<td>21.7385</td>
<td>TourismService</td>
<td>Travel_information</td>
<td>orionWestGreece-UNIFI</td>
</tr>
<tr>
<td>38.2384</td>
<td>21.7385</td>
<td>TourismService</td>
<td>Travel_information</td>
<td>orionWestGreece-UNIFI</td>
</tr>
<tr>
<td>38.2384</td>
<td>21.7385</td>
<td>TourismService</td>
<td>Travel_information</td>
<td>orionWestGreece-UNIFI</td>
</tr>
<tr>
<td>38.2384</td>
<td>21.7385</td>
<td>TourismService</td>
<td>Travel_information</td>
<td>orionWestGreece-UNIFI</td>
</tr>
<tr>
<td>38.2384</td>
<td>21.7385</td>
<td>TourismService</td>
<td>Travel_information</td>
<td>orionWestGreece-UNIFI</td>
</tr>
<tr>
<td>38.2384</td>
<td>21.7385</td>
<td>TourismService</td>
<td>Travel_information</td>
<td>orionWestGreece-UNIFI</td>
</tr>
<tr>
<td>38.2384</td>
<td>21.7385</td>
<td>TourismService</td>
<td>Travel_information</td>
<td>orionWestGreece-UNIFI</td>
</tr>
</tbody>
</table>
Variables of the Device

- `dateObserved` for Time Series, UNIQUE!

- Variables have to be assigned to:
  - Value Type, Value Unit, Data Type

- For example for Device

  BasicSizesofIncomingTourismoftheRegionofWesternGreece.xlsx__UnitedKingdom

<table>
<thead>
<tr>
<th>Region</th>
<th>Visits in Thousands</th>
<th>receipts in Millions Euro</th>
<th>Nights in Thousands</th>
<th>Expenditure per Visit Euro</th>
<th>Cost per Night Euro</th>
<th>Average Length of Stay</th>
<th>dateObserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>West Greece</td>
<td>1115</td>
<td>12.8</td>
<td>156.6</td>
<td>165.3</td>
<td>774</td>
<td>21</td>
<td>2019-12-31 00:00:00</td>
</tr>
<tr>
<td></td>
<td>183</td>
<td>18.8</td>
<td>183.8</td>
<td>142.3</td>
<td>128.3</td>
<td>14</td>
<td>2019-12-31 00:00:00</td>
</tr>
<tr>
<td></td>
<td>1187</td>
<td>29</td>
<td>222.9</td>
<td>209.2</td>
<td>133.1</td>
<td>16</td>
<td>2019-12-31 00:00:00</td>
</tr>
<tr>
<td></td>
<td>1213</td>
<td>28.6</td>
<td>226.8</td>
<td>199.7</td>
<td>175</td>
<td>17</td>
<td>2019-12-31 00:00:00</td>
</tr>
<tr>
<td></td>
<td>462</td>
<td>17.4</td>
<td>3725</td>
<td>3626</td>
<td>46.0</td>
<td>19</td>
<td>2019-12-31 00:00:00</td>
</tr>
<tr>
<td></td>
<td>497</td>
<td>20</td>
<td>293.5</td>
<td>462.5</td>
<td>88.8</td>
<td>5.8</td>
<td>2019-12-31 00:00:00</td>
</tr>
<tr>
<td></td>
<td>428</td>
<td>113</td>
<td>166.6</td>
<td>377.6</td>
<td>74.9</td>
<td>4.2</td>
<td>2019-12-31 00:00:00</td>
</tr>
<tr>
<td></td>
<td>477</td>
<td>17.9</td>
<td>344.8</td>
<td>373.2</td>
<td>91.8</td>
<td>7.2</td>
<td>2019-12-31 00:00:00</td>
</tr>
<tr>
<td></td>
<td>425</td>
<td>13.6</td>
<td>279.9</td>
<td>392.2</td>
<td>57.0</td>
<td>5.6</td>
<td>2019-12-31 00:00:00</td>
</tr>
</tbody>
</table>
What May Happen Later

Initial Automated Activities

A) User may Add data on IoT Directory Tool (time series)

B) IoT Device may send additional data (time Series) on Broker

C) Other IoT Apps may get / produce and load additional Data (time Series)
Notes on the implementation

• The Data Table Loader has been developed in PHP to interact with the user to regularize data in ingestion, request missing information, etc., and finally to save this information on a **DataTable** in MySQL
  – A Status for data ingestion is defined, managed, evolved

• **The IoT App** gets the data and when possible and needed: creates the IoT Device Modes, IoT Devices, and IoT Device Instances (time series)

• Any AreaManager can upload DataTable sets but only specific dedicated responsible users for data upload can actually load. We suggest one.
  – Each of them is becoming the owner and responsible of the IoT App process, which can be customized also, and of the IoT Device Model, IoT Device.

Snap4City (C), January 2022
IoT Devices for Events
• May be sporadic and periodic
• May be events associated with the same device (Repeat), or the device can have a single event (OneShot)
• They have GPS, DeviceID, dateObserved, etc.....may be mobile or fixed

• **OneShot events**
  – Landslide, incident,

• **Repeat events**
  – Entertainment representation, ticket of a telepass, ticket of a speed taxation fixed position, maintenance ticket of plant or of a car, etc.
### Coded Variables for Event Management

<table>
<thead>
<tr>
<th>variables</th>
<th>description</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>eventDateStartShow</td>
<td>Even if the event start at evenDateStart it should be shown since this date</td>
<td>20-05-2021T.......</td>
</tr>
<tr>
<td>eventDateStart</td>
<td>Start of the event</td>
<td>06-06-2021T.......</td>
</tr>
<tr>
<td>eventDateEnd</td>
<td>End/closure of the event, from start to even a number of states can be possible</td>
<td>07-06-2021T.......</td>
</tr>
<tr>
<td>eventSeverity</td>
<td>from 1 to 100, for coded severity or critical, 1 non critical, 100 very critical</td>
<td>1</td>
</tr>
<tr>
<td>eventType</td>
<td>ONESHOT</td>
<td>REPEAT</td>
</tr>
<tr>
<td>eventUniqueIdentifier</td>
<td>Valid only if eventType==REPEAT</td>
<td>2DD8342349529DS</td>
</tr>
<tr>
<td>eventKind</td>
<td>TICKET, ENTERTAINMENT, SLIDELAND, ACCIDENT, ICE, NATURALDISASTER, BOMB, ROADCLOSURE, MARKET, WATERFLOoding, PLANNED, MAINTENANCE, UNEXPECTED, UNKNOWN, FIRE, etc.</td>
<td>TICKET</td>
</tr>
<tr>
<td>eventStatus</td>
<td>Some coded status, depending on the event kind: START, END are mandatory</td>
<td>PAID</td>
</tr>
<tr>
<td>eventColorStatus</td>
<td>Color of the status, depending on the event kind: START, END are mandatory</td>
<td>4</td>
</tr>
<tr>
<td>eventShownStatus</td>
<td>Status to be shown, depending on the event kind: START, END are mandatory</td>
<td>PAID</td>
</tr>
<tr>
<td>eventIconID</td>
<td>Icon to be shown in current status, depending on the even kind</td>
<td>https://...../paidicon.png</td>
</tr>
<tr>
<td>eventDescription</td>
<td>Textual description of the event and its evolution</td>
<td>A paid ticket</td>
</tr>
<tr>
<td></td>
<td>A set of specific Variables</td>
<td>.................</td>
</tr>
</tbody>
</table>
Ingesting Public Transport Information
Public Transport Information/file/streams

- used for: busses, train, ferry, metro, tramways, etc.
- Include:
  - Public Transport Lines, Rides with paths and timeline, stops, polylines for paths, etc.
  - real time data about the position of the vehicles: train, busses, etc.
  - Multi operator data
- Information is modelled as
  - GTFS format: multiple files in XML
  - Transmodel format
- GTFS files can be ingested on Snap4City via
  - ETL which takes GTFS files and convert them in triples «.n3» file for the Knowledge Base
    - https://github.com/disit/smart-city-etc/tree/master/TrasformazioneTPLBus_new_model/Triplification/Models
    - Former version: https://www.snap4city.org/download/snap4cityETL/TPL_bus_gtfs/
  - Chouette and then
    - using a Python developed by Snap4City to converter to produce Triples for the Knowledge Base, service map
      - https://github.com/disit/snap4city/blob/master/Snap4CityGTFS/chouette-gtfs-n3.py
- Transmodel (EN12896) or Neptune files can be ingested in Snap4City via
  - Chouette and then, with a certain level of adaptation,
    - using a Python developed by Snap4City to converter to produce Triples for the Knowledge Base, service map
      - https://github.com/disit/snap4city/blob/master/Snap4CityGTFS/chouette-gtfs-n3.py
• Interoperable with: GTFS, Transmodel, Neptune and «NeTEx»

http://www.chouette.mobi/en/
Verification of Data Ingestion
Snap4City, Snap4Industry Architecture, V2

**Data Sources, External Services**
- Pull Data

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

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

**Knowledge base**
- Semantic Reasoners

**Indexing and aggregating**
- OpenDistro x Elastic Search

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

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

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

**Authentication, Authorization, Platform & Processes Management, Data Inspector, Digital Twin, ...**

**Inform, announce, Act!, warning, alarms, What-If, ...**

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

Snap4City (C), January 2022
Snap4City, Snap4Industry Architecture, V2

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

Data Ingestion, aggregation, regularisation, reconcile: IOT Directory, NIFI, special tools

Knowledge base
- Semantic Reasoners

Indexing and aggregating
- OpenDistro x Elastic Search

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

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

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

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

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

Authentication, Authorization, Platform & Processes Management, Data Inspector, Digital Twin, ...

Snap4City (C), January 2022
Smart City Functional Architecture

- Transport systems
  - Mobility, parking

- Public Services, Govern, events, ...

- Sensors, IOT Cameras, Wi-Fi

- Environment, Water, energy

- Shops, services, operators

- Social Media
  - Crawler and Manager

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

Data and channel interface interoperability

Big Data Storage and Indexing

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

Dashboards, visual tools, Web and Mobile Apps
- Back office tool
- Front-End
- Back-End

Data Analytics, Simulations, Special Tools
- R Studio, TensorFlow, Python, ...

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

Inform, announce, Act!, warning, alarms, What-IF
- Authorization, Platform & Processes Management, Data Inspector, Digital

Snap4City (C), January 2022
**Smart City Functional Architecture**

- **Transport systems**
  - Mobility, parking

- **Public Services**
  - Govern, events, ...

- **Sensors, IOT Cameras**, Wi-Fi

- **Environment, Water, energy**

- **Shops, services, operators**

- **Social Media**
  - Social Media Crawler and Manager

---

**Data Sources, External Services**
- PULL Data

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

---

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

---

**Knowledge base**
- Semantic Reasoners

**Indexing and aggregating**
- OpenDistro x Elastic Search

---

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

---

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

---

**Data Analytics, Simulations, Special Tools**
- R Studio, TensorFlow, Python, ...

---

**Back-End**

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

**Dashboards, visual tools, Web and Mobile Apps**

---

*Snap4City (C), January 2022*
Main IoT Data In/Out flows

- IOT Orion Broker
- Knowledge Base
- Semantic Reasoners
- Indexing and Aggregating
- NIFI, OpenDistro per ElasticSearch
Which are the mechanisms to send data into the Open Distro Elastic Search?

The in place mechanisms to send data on elastic search are

- (i) passing from an IOT Broker registered on IoT Directory, and thus directly registered on NIFI which will be registered to all the events arriving on the Broker,
- this feeding is automated if the IoT device (physical or virtual) is registered on IoT Broker and thus on IoT Directory and NIFI
- (ii) providing data as MyKPI deciding their storage on ES at level of root admin, also in this case, the feeding is automatic
- (iii) automatically from Data Table Loder that takes info and data from an excel file and generates IoT Devices and data, creating automatically data model, registered IoT Devices, and time series

In all cases, the feeding can be performed via IoT App, or via API, or Manually generate the single instance. For IoT Devices via IoT directory, and for MyKPI, via MyKPI manager.
Checking data ingestion results

Knowledge base
Semantic reasoners
- All searches
- Metata
- Structure
- Last values of IOT Dev
- GTFS
- Only public IOT Dev

Indexing and aggregating
OpenDistro x Elastic search
- Faceted search
- Geo search
- Time Series
- Private and Public

- Data Inspector
- ServiceMap, SCAPI
  - LOG / LOD viewer
  - Super Service Map
- IOT Directory
- SCAPI: Swagger
- IOT Broker

- Data Inspector
- ServiceMap, SCAPI
- My Data Dashboard
  (Kibana), DevDash
- OpenDistro x Elastic Search

Some functionalities are limited to certain roles
Verification of Data Ingestion Process

• Verify that
  – IOT Device Creation see it on
    • IOT Directory, IOT Devices list
    • Service Map if the device is Public
    • Data Inspector if the device is public and/or private
      – You can see the trend to see the time series and last value from the pin on map of the Data Inspector
    • Dashboards...... you can create a specific one using the Dashboard Wizard
  – IOT Device Data (time series) see them on
    • IOT Broker, via ...... A dedicated services is coming....
    • Service Map if the device is Public
    • Data Inspector, this means that the data are on Storage
      – You can see the trend to see the time series and last value from the pin on map of the Data Inspector
    • Dashboards...... you can create a specific one using the Dashboard Wizard
1. Verify the presence of the IoT Device you created

2. Verify the structure of the device by edit tool

3. See the NGSI V2 JSON format to be used on sending data msgs

4. Call the IoT Broker to see the last data on it (some user name and Pwd can be needed)
   - Get the Broker Name from the device profile

5. See next slide

Broker service URL such as
-- https://www.snap4city.org/brokername/v2/entities
-- http://brokername.snap4city.org/v2/entities/
Verify on Knowledge Base

5. Click on ServiceURI (device URI) to Open in a new TAB the data sent on the KB

6. If your device is Public and you have sent data → the list of the last data from OpenDistro x Elastic Search querying from KB will appear as:

Snap4City (C), January 2022
Very data on KB via ServiceMap

- **7)** Verify on ServiceMa by Search on data location or by text name of the device
- **8)** click on ServiceURI to jump on LOG.DISIT.ORG to see semantic structure

Snap4City (C), January 2022
Very data on ES Kibana

- Verify on OpenDistro x ElasticSearch on Kibana, My data Dash:
The IOT Orion Brokers can be feed by means of:
- IOT App of Snap4City (to implement Agents and/or Adapters)
- IOT Agents and/or NGSI Adapters by FiWare for different protocols
- IOT Brokers of any kind, different protocols and producers, also as Gateways, and they can be located on premise and/or on any cloud

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

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

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

**How to access at SURI**

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

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

**See Part 3**
Usage of the ServiceMap and Knowledge Base Browsing
Knowledge Base
Semantic Reasoners
• **Km4City is the reference ontology for Snap4City**, it allows to:
  – keep connected city entities each other:
    • Semantic Index, reticular
    • Perform spatial, geographic, and temporal reasoning
  – Discover city entities and their relationships via IOT App and Smart City API:
    • IOT devices, IOT sensors, city elements, roads, services, Brokers, etc.
  – Provide access via Advanced Smart City API
  – Federate other Km4City Knowledge Bases, the approach allows to scale geographically and create redundancies, improving performances

• **Documentation**
  – **TC5.15 - Snap4City Smart City API Collection and overview, real time**
  – **ServiceMap and ServiceMap3D, Knowledge Model, Km4City Ontology**
  – **Knowledge Base Graphs and Queries: browsing and queries into the KB**
Road Graph (Tuscany region)
132,923 Roads, 389,711 Road Elements
318,160 Road Nodes, 1,508,207 Street Numbers
Info on: points, paths, areas, etc.
Services (20 cat, 512 cat.)
16 Public Transport Operators
21,280 Bus stops & 1,081 bus lines

Dynamic/real-time as in Tuscany Region
• Real time bus lines: 144 updates X day X line
• 1,081 Transport Pub Lines: 1-2 up per day, time-path
• >210 parking lots status: 76 updates X day X sensor
• >796 traffic Sensors: 288 updates X day X sensor
285 weather area: 2 updates X day X area
>12 hospital Triage status: 96 updates X day X FA
• 600 Environmental data: 20 updates X day X sensor
• 39 Bike Sharing racks data: Pisa and Siena
• 12 Pollution data, 37 air quality data
• 177 recharging stations
• Smart benches, waste mng, irrigators, lighting, ...
• Florence ent.events: about 60 new events X day
Different kinds of Florence traffic events,
[1,600 Fuel stations: 1 update X day X station]
[Wi-Fi: > 400,000 measures X day]
• App mobiles: >50,000 measures X day
more than 40,000 distinct users X day
• From 600,000 to 4.5 M Tweets X day
• + many IOT sensors personal and industrial ...
• ServiceMap is the **main Tool** to:
  – monitor the status of the Knowledge Base
  – test queries and produce query and SmartCity API testing calls for developers
    • Any kind of search (semantic, full text, etc.), routing,
  – Access at the specific Graph Data base via LOG.disit.org

• **ServiceMap is showing:**
  – only **public data**. Private data are not shown via ServiceMap but can be accessed via DataInspector
  – data regarding **a single Knowledge Base** of the federated network of KBs. Each KB may contain multiple Organizations.
  – technical views for developers
**Discovery**

**Search by Shape and Distance**

Each request or search in the Km4City model can be referred to a point and a ray, to an area, to a polyline.
Cycling Paths

ServiceMap: https://servicemap.km4city.org
Views of the Knowledge Base

- How pass from ServiceMap to Linked Open Graph, Linket Data view tool
Main Organizations/areas

- Antwerp area (Be)
- Capelon (Sweden)
- 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)
Km4City Federation

At different levels:

– Among cities/regions
– Among data providers
– Among Operators

By Means of:

– Smart City API → Apps
– Km4City Smart City Ontology
– Dashboards/data analytics
Federation of Snap4City Services

- A Mobile App may refer to one Smart City API Server (for Area 1) via SUPER and receive data from the Federated SUPERS (Area 2) if navigation, queries, etc. are leading to discover out of the addressed KB.
  - SUPER can be used for creating redundant and/or balanced distributed solutions for Federated KB. See Area 2, the two KB in the front.
  - Federated SUPER ServiceMap can have overlapped KB even totally.
  - A Mobile App can be developed to support multiple Smart City API servers, for balancing and
- The usage of Super (ServiceMap) is not mandatory so that separate services can be produced as well.
  - SuperServiceMap and ServiceMap presents the same Smart City APIs.
• Super, Nodes and SSM2ORION presents the same Smart City APIs.
• The network of Super can be reconfigured dynamically
  – Multiple networks of Super can be realized as well
  – Distributed Searches via the Federation of Super are performed with o(1) complexity
  – Results from an API rest calls are provided in real time also when the size of the network is large
  – Dashboard widgets and Mobile Apps are enabled to use the Super
  – Clients can pass from one Super to another transparently: moving devices
• Nodes
  – do not need to permanently share data
  – data can be of any size, the data shared is typically public since users of different KB are different and not refer to the same LDAP/KeyCloak authentication/authorization service.
  – may have different number of services
  – Services can be based on KB as well as on Brokers
  – Services managed as HLT of: Sensors, Sensor-Actuators, POI.
    • Data of other HLTs are managed independently from the other SmartCity API such as: MyKPI, External Services, WFS GIS, Heatmaps, special tools, etc. etc.
• The solution support disjoined nodes, federation and independent services
Federated ServiceMap and Smart City API

To improve scalability, fault tolerance and federation among cities:

- One entry point Smart City API for all zones
- Multiple Knowledge base See performance assessment

At different levels:

- Among cities/regions
- Among data providers, Operators

By Means of:

- Smart City API → Apps
- Smart City Ontology
- Dashboards/data analytics
- Organization independent
Snap4City IoT Registration and Access

A range of IoT Brokers and protocols

A range of other data sources

MySQL

Km4City

Data Connectors

IoT Directory

ServiceMap

Smart City API

Authentication and Authorization

Dashboard Builder

Mobile Apps

Dashboards
Federation of Snap4City vs IOT ORION Broker

Federation of Snap4City vs IOT ORION Broker

Hybrid Solutions

Snap4City Solutions

Area 3

IoT Agent

Crate-DB

MongoDB

Quantum Leap

IoT Orion Broker

4200:4200

8668:8668

27017:27017

4041:4041

8668:8668

1026:1026

NGSI

NGSI

SSM2ORION

SUPER

Dashboard Builder

FIWARE
Some functionalities are limited to certain roles
Context and Problems

• Smart city context includes solutions that presents
  – **Data**: heterogeneous, large volume, several protocols, legacy data systems, several semantics, real time, multiple domains, etc..
  – **Processes**: several of different kinds, aperiodic, periodic, event driven, ....
  – **Users**: owners, responsibilities, developers, final users, etc.
  – **Relationships**: among data, processes, users and mixt
  – **Non Functional Requirements**: security, GDPR, reliability, quality, scalability, etc.
    - Interoperability: legacy, protocols, modularity,

• **Data Ingestion**: the models and mechanism for data gathering
• **Data Inspection**: the model and solution to identify problems, and understand solution
Data Inspector (Digital Twin info) Major Submodels

• Digital Twin
  – Device and sensors data
  – Values
  – Healthiness criteria and values
    • Machine learning tools
  – Images and physical world
  – Licensing
  – Users

• Users
  – Defined the Data and Devices
  – Defined the processes
  – Create dashboards
  – Etc.

• Process Views
  – Device Management tool
  – Data ingestion processes
    • ETL, IOT Apps
  – Data storage access views
    • Index views
    • Relationships view
  – Data Analytics and Transformation
    • IOT App, R Studio, Python
  – Data Rendering Dashboards
    • Synoptics
  – Processes’ Developers
Data Inspector: all you need to know about data, data sources and ingestion processes
• It is showing data:
  – Listed by High Level Types, and classified as in the Dashboard WIZARD: see Tutorial Part 2 for details.
  – of your Organization only, that are public of the organization, your private or those that have been delegated to you some how.
Data Inspector (open on your left side menu)

- Cross Filtering on the basis of:
  - **MAP**: pan and zoom → lock, center on GPS coordinates of the user
  - **Data Source Classification**:
    - faceted filtering
    - full text search
  - Click on data source to see it on map, and see the graphics representation, just to learn how widgeting it.
  - **Full Text Search** if you remember some desc...

- Selecting a Data Source on Map: on its Pin you can see:
  - Real time data
  - Time trend: 4 hours, 12 hours, 1 day, 1 week, 30 days.
  - Full status and description (only for Administrators)
Click with the mouse on it

Some functionalities are limited to certain roles
• Specific values of selected
• Information of the values of the other sensors on the same device
• View Trends, marking problems, healthiness by point according to a Fuzzy model
• Marking problems for future machine learning processes (separate tool)

Some functionalities are limited to certain roles
### Healthiness

- **HLT: Sensor**

<table>
<thead>
<tr>
<th>Device</th>
<th>Values</th>
<th>Healthiness Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value Type</td>
<td>meanPeople</td>
<td></td>
</tr>
<tr>
<td>Healthiness Criteria</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delay</td>
<td>81347</td>
<td></td>
</tr>
<tr>
<td>Data Type</td>
<td>float</td>
<td></td>
</tr>
<tr>
<td>Period</td>
<td>900</td>
<td></td>
</tr>
<tr>
<td>Last Update</td>
<td>2020-07-10T13:06:34.734+02:00</td>
<td></td>
</tr>
<tr>
<td>Healthiness Criteria 1</td>
<td>(2020-07-19 23:03:31)</td>
<td>false</td>
</tr>
<tr>
<td>Healthiness Criteria 2</td>
<td>(2020-07-19 23:03:31)</td>
<td>false</td>
</tr>
</tbody>
</table>

Some functionalities are limited to certain roles.

- **Two different criteria**
  - **H1**: at least an event in the last 24 hours
  - **H2**: machine learning for most of Sensors devices
Details regarding the IOT Ingestion process

- For IOT Device data
- IOT Broker details

Some functionalities are limited to certain roles
Details regarding the Ingestion process

- For ETL processes
- Scheduling details and status

Some functionalities are limited to certain roles

---

Snap4City (C), January 2022
Some functionalities are limited to certain roles
HLT: From Dashboard to IOT APP

- Click with the mouse on it
HLT: Dashboard-IOT App (From Dashboard to IOT APP)
HLT: Sensor-Actuator (From Dashboard to IOT Device)
Some functionalities are limited to certain roles.
The fields shown may be present or not depending on the HLT and on the information received.
Report Generation and Access

• **Device/data** owner may have their reports: monthly or 3-monthly
  – Ready to use reports are available for:
    • Single Device: ETL and IOT
  – Ask to your RootAdmin to activate the production of reports (and also **hourly** report for testing only).

1. Open data Inspector
2. Click on Device or sensor
3. Click on report
4. Get the Last Report
Data Inspector recent and future features

• Specific views for the HLT
• Details on Healthiness
• More details on the Ingestion Process
• Capabilities of setting and changing the Healthiness criteria
• For data related to IOT App vs Dashboard, how the link to them
• A view about the data relationships, precisely to show the data used in dashboard, and used in IOT App, etc.
• For Sensors: a link to DevDash to see the time trend and relationships with other sensors and devices on the same Organization (for all in the case of RootAdmin)
• A reverse link from the SmartAppGraph to the Data Inspector data of various kind
DevDash based assessing Elastic Search (OpenDistro) Data via Kibana

Some functionalities are limited to certain roles
DevDash: My Data Dashboard Kibana
Business Analysis Dashboards
For all kind of users: DevDash

• Dynamic Filtering, Adaptable, ...
• Full data details, drill down,...
• Synergic with Data Inspector which addresses data relationships, processing and information
• Only Your Data for
  • Manager and Area Managers
• All Accessible Data for
  • ToolAdmin and RootAdmin
• Multi facetted Search by
  • Devices
  • Organization
  • Drill on Time
  • Drill on Map
  • Value Types
  • Data Type
  • Value name
  • Data table
  • Etc.

• Respect Privacy and GDPR
• My Data Dashboards (for Developers and for Managers) can be customized by RootAdmin.
  – Authority for Customization can be also extended to other role on Premise solutions
• Customizations is based on Full editing Capabilities of OpenDistro x Elastic Search 7.1 and Kibana, and with multiple indexes of Snap4City
Technical Ingestion of Structural Data (road graph and POI)
Set up of the Road Graph on Knowledge Base, performed with an open source tool
Snap4City Architecture vs Data Ingestion

**Data Sources, External Services**
PULL Data

**Knowledge base**
Semantic reasoners

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

**Rendering**
Acting, Widgets, MicroApps
User interface, Interactive Dashboard, Drill down, maps, heatmaps

**Inform, announce, Act!, warning, alarms, What-IF,** ..
Km4City: Knowledge Base

- Multiple DOMAINS
- Geospatial reasoning
- Temporal reasoning
- Metadata
- Statistics
- Risk and Resilience
- Licensing
- Open and Private Data
- Static and Real time
- IOT/IOE

 Ontology Documentation:
http://www.disit.org/6506
http://www.disit.org/6507
http://www.disit.org/5606
http://www.disit.org/6461

- Street-Guide
- Mobility and transport
- Points of interest
- Sensors, IOT, ...
- Energy
- Administration
- Citations from strings

Big Data Tools

Ontology Documentation:
http://www.disit.org/6506
http://www.disit.org/6507
http://www.disit.org/5606
http://www.disit.org/6461

Schema: http://www.disit.org/km4city/schema
RDF version: http://www.disit.org/km4city.rdf
Smart-city Ontology: 1.6

- covers different aspects:
  - Administration
  - Street-guide
  - Points of interest
  - Local public transport
  - Sensors
  - Temporal aspects
  - Metadata on the data
  - Industry 4.0 structures
Smart-city Ontology km4city

License Free
1.6.7

Also covers Industry 4.0 structures

https://www.snap4city.org/19
Km4City Ontology elements 1.6.7

- **Km4C**: Km4City 1.6.7
- **Using**
  - **DCTERMS**: for metadata Dublin Core Metadata Initiative
  - **FOAF**: friends of a friends
  - **Good Relation**: entities relationships
  - **iot-lite**: IOT Vocabulary
  - **OTN**: Ontology of Transportation Networks
  - **OWL-Time**: time reasoning
  - **SAREF**: Smart Appliances REFerence extension for building devices available at [https://saref.etsi.org/saref4bldg/](https://saref.etsi.org/saref4bldg/)
  - **Schema.org** for people and organizations
  - **SSN**: Semantic Sensor Network Ontology (see [https://www.w3.org/TR/vocab-ssn/](https://www.w3.org/TR/vocab-ssn/))
  - **WGS84**: Datum of Geo-Objects
  - **GTFS**, General Transit Feed Specification, and **Transmodel**, for public transport infrastructures: time schedules, real-time records, paths, etc.;
Set up of the Knowledge Base, KB

- The **KB starts empty**, it has to be initialized with the Road Graph(s) of interest.
- **Road Graphs** can be obtained from:
  - GIS of the municipalities, regional govern, etc.
  - Open Street Map, OSM
  - Etc.
- See this note on KM vs OSM: [https://www.snap4city.org/397](https://www.snap4city.org/397)
- Snap4City provides a tool for feeding the KB with OSM
  - TC5.10- Open Street Map ingestion process
  - [From the Open Street Map to the Km4City street graph](https://www.snap4city.org/download/video/From%20the%20Open%20Street%20Map%20to%20the%20Km4City%20street%20graph.pdf)
  - OSM2KM4C tool is included into KBSM, VM and Docker [https://www.snap4city.org/471](https://www.snap4city.org/471)
  - Tool: [https://github.com/disit/osm2km4c](https://github.com/disit/osm2km4c)
- The load of a city of 1 Million of inhabitants can be done in few hours.

Snap4City (C), January 2022
Snap4City Data Ingestion Flow Diagram

Static or quasi Static Data: POI, etc.

IOT Device Model Registration

IOT Device Registration

Real Time data in Pull
Any gateway, server

periodic

IOT Device

push

pull

sporadic

(1c) static

IOT App

IOT Directory

Dynamic

IOT App

IOT Orion Broker

IOT Orion Broker

push

pull

NGSI

IOT App

Adapter

Information, File

IOT Brokers

IOT Devices

push

push

push

push

Automatic

IOT Brokers

IOT Devices

push

Knowledge Base
Semantic Reasoners

Indexing and Aggregating
NIFI, OpenDistro x ElasticSearch

subscription note

periodic

OSM to Road
Graph Setup

tripples

SURI Link

Snap4City (C), January 2022
Which are the mechanisms to send data into the Knowledge Base?

The KB is feed with new concepts and entities, and they are produced by the solution and feed into the KB:

– (i) automatically by the IoT Directory about all the new registered IoT Devices which are registered on some IoT broker which have been already registered on IoT Directory
– (ii) automatically from an IoT App it is possible to generate triples in somehow and post them in N3 formats into the specific KB you targeted, according to the ORG you have
– (iii) automatically from POI Loader tool that takes Excel file in input and generate triples for a specific your organization
– (iii) manually producing triples and send them to RootAdming for feeding KB, or using an IOT App for feeding KB
– (iv) converting OSM in triples about road graph details by using a tool called
  • From the Open Street Map to the Km4City street graph
  • [https://www.snap4city.org/download/video/From%20the%20Open%20Street%20Map%20to%20the%20Km4City%20street%20graph.pdf](https://www.snap4city.org/download/video/From%20the%20Open%20Street%20Map%20to%20the%20Km4City%20street%20graph.pdf)
Linked Open Data

LOG: https://log.disit.org

Schema: http://www.disit.org/km4city/schema
RDF version: http://www.disit.org/km4city.rdf
Linked Open Data

LOG: https://log.disit.org
Programming Static Data Ingestion via IOT Apps, API, and Files
Snap4City Data Ingestion Flow Diagram

Static or quasi Static Data: POI, etc.

IOT Broker Registration

IOT Device Model Reg.

IOT Device Registration

Real Time data in Pull
Any gateway, server

IOT Device

IOT App

IOT Directory

Knowledge Base
Semantic Reasoners

Indexing and Aggregating
NIFI, OpenDistro x ElasticSearch

IOT Orion Broker

IOT Brokers

IOT Apps

Snap4City Tools

IOT Device/Gateway

Information, File

SURI Link

subscription note

Real Time

pull

push

periodic

pull

push

sporadic

(1c) static

Dynamic

sporadic

IOT Orion Broker

IOT App

Adapter

Adapter

NGSI

NGSI

Indexing and Aggregating
NIFI, OpenDistro x ElasticSearch

Snap4City (C), January 2022
Snap4City Data Ingestion Flow Diagram

Static or quasi Static Data: POI, etc. pull sporadic

IOT App

Knowledge Base
Semantic Reasoners

Registration triples

Information, File
IOT Apps
Snap4City Tools
IOT Device/Gateway

Indexing and Aggregating
NIFI, OpenDistro x ElasticSearch

SURI Link
Regularizing Static data via IOT App

Pre-requisites:
• Knowledge Base main categories (Service Map or)

Register Static Data on Snap4City KB:
1. ‘Regularize’ your data putting them in csv file (possible fields available here: https://www.snap4city.org/589)
2. Create your IoT App starting from the sample available here:
   – Description: https://www.snap4city.org/596
3. Upload your csv containing N rows, each row is a Point of Interest
Regularization

1. Create your csv (with 33 fields in the header). Here after some of them (M = to be filled Mandatory):
   - id: Unique POI identifier (Recommended, if not present is created starting form the file name)
   - subCategory (M): KM4City category (ex: "Museum")
   - latitude (M): POI latitude (ex: "43.7936")
   - longitude (M): POI longitude (ex: "11.2604")
   - province: (ex.: "FI")
   - nameITA/nameENG: POI name
   - city: (ex.: "FIRENZE")
   - postalcode: (ex.: "50100")
   - streetAddress: (ex.: "via di Santa Marta")
   - civicNumber: (ex.: "3")
   - descriptionShortENG : (ex: "Exhibition... ")
   - url: (ex: "https://www.snap4city.org")
   - … others...here: https://www.snap4city.org/589
<table>
<thead>
<tr>
<th>id</th>
<th>subCategory</th>
<th>latitude</th>
<th>longitude</th>
<th>province</th>
<th>city</th>
<th>postalcode</th>
<th>streetAddress</th>
<th>civicNumber</th>
<th>nameITA</th>
<th>nameENG</th>
<th>description</th>
<th>url</th>
</tr>
</thead>
<tbody>
<tr>
<td>Museum_FI_history1</td>
<td>Museum</td>
<td>43.7936</td>
<td>11.2604</td>
<td>FI</td>
<td>FIRENZE</td>
<td>50127</td>
<td>Via Mazzini</td>
<td>34</td>
<td></td>
<td></td>
<td>Museum storia contemporanea; Museum of Contemporary History</td>
<td></td>
</tr>
<tr>
<td>Theatre_FI_xx</td>
<td>Theatre</td>
<td>43.6897</td>
<td>11.2587</td>
<td>FI</td>
<td>FIRENZE</td>
<td>50141</td>
<td>Piazza Garibaldi</td>
<td>4</td>
<td>Teatro Comunale; Municipal Theatre; Funded by ...</td>
<td>https://www.teatro_123.com</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hotel_FI_yyy</td>
<td>Hotel</td>
<td>43.6572</td>
<td>11.2596</td>
<td>FI</td>
<td>FIRENZE</td>
<td>50153</td>
<td>Via s. Marta</td>
<td>13</td>
<td>Hotel Nuovo; New Hotel; offerte ...</td>
<td>https://www.hotel_xxx.com</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
IoTApp

Create your IoTApp:

- Login > IoT Application > ’Create New’ button
- Wait some minutes
- Copy the Source code available here: https://www.snap4city.org/drupal/sites/default/files/files/iotapp-poi-ingestion.txt
- Import the code and save (‘Deploy’ button)
Modify your IoTApp:

- Open the ‘get simple block’ and search for your App id (ex: ‘nr8jero’)
- Put your mail and credentials on the ‘send email form ..’ block
- Go to the page ‘https://iot-app.snap4city.org/nodered/nr8jero/simple’ and upload your csv file
- Add your e-mail credential

https://iot-app.snap4city.org/nodered/nr8jero/simple

This page is a service for generating triples from CSV files of POI. Please upload a CSV file according to the instructions of page https://www.snap4city.org/589: you are going to receive an email with a file to be loaded in your KB.
This page is a service for generating triples from CSV files of POI

please upload a CSV file according to the instructions of page https://www.snap4city.org/589:
you are going to receive an email with a file to be loaded in your KB:

![File upload interface](image)
Modify your IoTApp:

- Send the file you receive to us, or copy in a file the triples available in the web page and send them us

  • Reference people:
snap4city@disit.org

File Snapcity_static_POI.csv uploaded!

Result sent to michela.paolucci@unifi.it

https://iot-app.snap4city.org/nodered/nr8jero/upload
How to load triples into the KB
(Admin and on-premise)
Loading Static Data into KB

• The loading is performed via IOT App, with a specific IOT App flow exploiting a MicroService/API of ServiceMap, KB
• Moreover for massing loading of Triples, also files of triples can be used posting them on a shared folder of the ServiceMap, KB
  – This is used for batch loading of triples
Load Triples in KB

Pre-requirements:
• Available only for your IOT App on premise and for administrators

Load Static Data on Snap4City KB:
• Register Static Data on Snap4City KB (point 1 and 2 on the right)
• Connect the additional blocks present in the IOT App and save (‘Deploy’ button)
• Upload your csv

https://www.snap4city.org/596

Register Static Data on Snap4City KB:
1. ‘Regularize’ your data (csv)
2. Create your IoTApp
3. Upload your csv
Load Triples in KB

- Additional Option Only for ONPREMISE Snap4City versions and for administrators
- In case you need to upload ONLY on the Knowledge Base a set of data containing static data and dynamic data all together:
  - Example: Cultural Events or weather predictions
- Use a predefined IoT App available here [https://www.snap4city.org/594](https://www.snap4city.org/594) (with the copy and past method, you can create your Mobile App)
- Add:
  - Inject block
  - Function block
  - Debug block
- The function block must contain …
- Now you can see the triples to be updated in the Knowledge Base in the Debug Tab
Technical Ingestion of Data
## Data Ingestion Strategy

<table>
<thead>
<tr>
<th>HLT</th>
<th>GPS &amp; Geo References</th>
<th>Info Meta Data</th>
<th>Variables Value Name</th>
<th>Time Serie, Seq.</th>
<th>In/Out Read/Write</th>
<th>Manual Ingestion Tool</th>
<th>Automated Loading tools</th>
<th>Technical IoT App management</th>
</tr>
</thead>
<tbody>
<tr>
<td>IoT Device Mobile</td>
<td>Yes Moving</td>
<td>yes</td>
<td>Multiple</td>
<td>yes</td>
<td>yes</td>
<td>Yes (IoT Directory)</td>
<td></td>
<td>Yes (IoT App..)</td>
</tr>
<tr>
<td>IoT Device</td>
<td>Yes only once</td>
<td>yes</td>
<td>Multiple</td>
<td>yes</td>
<td>yes</td>
<td>Yes (IoT Directory)</td>
<td>Yes (Data Table Loader)</td>
<td>Yes (IoT App..)</td>
</tr>
<tr>
<td>MyKPI</td>
<td>Yes Moving</td>
<td>yes</td>
<td>1</td>
<td>yes</td>
<td>yes</td>
<td>Yes (MyKPI editor)</td>
<td></td>
<td>Yes (IoT App..)</td>
</tr>
<tr>
<td>POI</td>
<td>yes</td>
<td>No</td>
<td>No</td>
<td>Write 1, read</td>
<td>No</td>
<td>Yes (POI Loader)</td>
<td></td>
<td>Yes (IoT App..)</td>
</tr>
<tr>
<td>Heatmaps</td>
<td>Area, matrix</td>
<td>yes</td>
<td>1 on the matrix</td>
<td>yes</td>
<td>Write 1, read</td>
<td>Load on Heatmap Manager</td>
<td>Yes via Rstudio, Python, IoT App</td>
<td>Yes (IoT App..)</td>
</tr>
<tr>
<td>Traffic Flow</td>
<td>roads</td>
<td>yes</td>
<td>1-4: traffic flow density, velocity,..</td>
<td>yes</td>
<td>Write 1, read</td>
<td>Load on Traffic Flow Manager</td>
<td>Special tools</td>
<td>Yes (IoT App..)</td>
</tr>
<tr>
<td>OD Matrix</td>
<td>Flow, mat</td>
<td>yes</td>
<td>1 on the matrix</td>
<td>Yes</td>
<td>Write 1, read</td>
<td>Load on OD Manager</td>
<td>Special tools, IoT App</td>
<td>Yes (IoT App..)</td>
</tr>
</tbody>
</table>
Snap4City, Snap4Industry Architecture, V2

Data Sources, External Services
- PULL Data
- Data Ingestion, aggregation, regularisation, reconcile: IOT Directory, NIFI, special tools

Knowledge base
- Semantic Reasoners
- Indexing and aggregating
- OpenDistro x Elastic Search

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

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

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

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

Back-End

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

Authentication, Authorization, Platform & Processes Management, Data Inspector, Digital Twin, ...

Snap4City (C), January 2022
Smart City Functional Architecture

Data Sources, External Services
- Pull Data
- Data Ingestion, aggregation, reconciliation, and semantic reasoning
- IoT Directory, NIFI, and special tools

Knowledge base
- Semantic Reasoners

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

Data Analytics, Simulations, Special Tools
- R Studio, TensorFlow, Python, ...

IOT Applications, Business Logic
- Node-RED, Snap4City Microservices

Authentication, Authorization, Platform & Processes Management, Data Inspector, Digital Twin, ...

Dashboards, visual tools, Web and Mobile Apps

Snap4City (C), January 2022
Snap4City Architecture vs Data Ingestion

- **Data Sources, External Services**
  - PULL Data

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

**Back-End**

- Knowledge base
  - Semantic reasoners

- Indexing and aggregating
  - OpenDistro x Elastic search

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

**Front-End**

- Rendering
  - Acting, Widgets, MicroApps

- User interface, Interactive Dashboard, Drill down, maps, heatmaps

**Data Ingestion, aggregation, regularization, reconcile:**

- NIFI, IOT App

**IOT Applications, Business Logic**

- Node-RED + Snap4City MicroServices

Inform, announce, Act!, warning, alarms, What-IF, ..
Data Ingestion Architecture and Processes
Snap4City Architecture vs Data Ingestion

Data Sources, External Services
- PULL Data

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

Back-End
- Knowledge base
  - Semantic reasoners
- Data Analytics, Simulations
  - R, TensorFlow, Python, ...
- IOT Applications, Business Logic
  - Node-RED + Snap4City MicroServices
- Indexing and aggregating
  - OpenDistro x Elastic search
- Search and Query, Smart City API
  - Facet, semantic search

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

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

- Created on browser
- A huge number of Protocols
  https://www.snap4city.org/65
- Scheduled internally and managed as Container
- Largely diffused approach as Node-RED
- Large number of Snap4City MicroServices, well documented
- PUSH and PULL models
- Simple mechanism to add new Features
- Very diffused in IOT
- Static and Dynamic data models depending on IOT Broker capabilities
- Scalable on Cloud
- Also present in IOT Edge devices

ETL processes

- Created with Spoon editor on VM (on premise or via remote access to VM)
- A Large number of protocols
  https://www.snap4city.org/65
- Scheduled by DISCES in the back office
- Well known data warehouse model
- Well documented for the process
- Only PUSH models
- Complex mechanism to add a new functionality
- Very diffused in Data transformation
- Static and Dynamic data models well linked ..
MyKPI vs IOT Device Sensor

- **MyKPI** can be grouped in a Groups of Values. They may have GPS changing.

- **Sensors** can be also grouped, but born directly grouped into an **IOT Device** as defined into the IOT Directory. They have GPS static.

- Please note that the "dateObserved" has to be in standard ISO to have Time Series.

IOT Device may be sending data as:

```json
{  "id":"dev45",  "type":"misura",  "attributes": [    { "name":"dateObserved",  "value":"2020-04-06T17:00:00.000Z",  "type":"time" },    { "name":"temp 1",  "value":28.976,  "type":"float" },    { "name":"Hum 1",  "value":35976,  "type":"float" },    ..........    { "name":"kevi",  "value":721732,  "type":"integer" }  ]}
```
## Data Ingestion Methods Comparison

<table>
<thead>
<tr>
<th></th>
<th>Datagate</th>
<th>ETL</th>
<th>IoTApp</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>types of data managed</strong></td>
<td>S</td>
<td>S, P</td>
<td>S, P, RT</td>
</tr>
<tr>
<td><strong>Data protocol types managed</strong></td>
<td>PULL</td>
<td>PULL</td>
<td>PULL and PUSH</td>
</tr>
<tr>
<td><strong>Scheduling</strong></td>
<td>external</td>
<td>external</td>
<td>internal</td>
</tr>
<tr>
<td><strong>Flows to manage N instances of the same dataset</strong></td>
<td>N</td>
<td>N</td>
<td>1</td>
</tr>
<tr>
<td><strong>Users’ technical level</strong></td>
<td>without</td>
<td>medium/high</td>
<td>low</td>
</tr>
<tr>
<td><strong>Development time</strong></td>
<td>1,2 hours</td>
<td>1, 2 weeks</td>
<td>3, 4 days</td>
</tr>
<tr>
<td><strong>Semantic (KM4City)</strong></td>
<td>standard template</td>
<td>ad hoc (manual)</td>
<td>ad hoc (semi-automatic)</td>
</tr>
<tr>
<td><strong>Developed number</strong></td>
<td>1334 datasets</td>
<td>162</td>
<td>76</td>
</tr>
<tr>
<td><strong>Mean number of blocks</strong></td>
<td>0</td>
<td>120.333</td>
<td>27.67</td>
</tr>
<tr>
<td><strong>Mean number of lines of code</strong></td>
<td>0</td>
<td>275</td>
<td>229</td>
</tr>
</tbody>
</table>
Snap4City Data Ingestion Flow Diagram

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

**Connections:**
- Pull: IOT App -> IOT Directory
- Push: IOT Orion Broker -> IOT Orion Broker
- Dynamic: IOT App -> IOT Directory
- Sporadic: (1c) static
- SURI Link
- Subscription note
- NGSI
- Indexing and Aggregating
- Information, File
- IOT Apps
- Snap4City Tools
- IOT Device/Gateways
Snap4city Data Ingestion Flow Diagram

- **Static Data:** POI, etc.
  - IOT Broker Registration
  - IOT Device Model Reg.
  - IOT Device Registration
  - IOT App
  - Knowledge Base
  - Semantic Reasoners

- **Dynamic Models:**
  - IOT Device
  - IOT Orion Broker
  - NGSI
  - FIWARE
  - NIFI, OpenDistro, x Elasticsearch

- **Real Time data in Pull:**
  - Any gateway, server
  - IOT App
  - IOT Orion Broker
  - NGSI

- **Periodic Data:**
  - IOT Devices
  - IOT App
  - Adapter

- **Sporadic Data:**
  - IOT Devices
  - IOT App
  - Adapter

- **Information, File:**
  - Snap4City Tools

- **IOT Device/Gateway:**
  - Adapter

- **Subscription note:**
  - NGSI
  - Real Time

- **SurI Link:**
  - Indexing and Aggregating

**Flow Diagram Elements:**
- Pull
- Sporadic
- Dynamic
- Periodic
- Push

**Snap4City (C), January 2022**
On the Flow Diagram

• **Static** (unified model for multidomain indexing)
  – Geodata, Open Data as POI, Data Bases, records, etc.
  – They change over time sporadically
  – See above POI Loader solution and IOT app

• **Models** (Registration of IOT Device Models, IOT Devices, IOT Brokers), for entities that have multiple instances in time / space
  – 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
HOW TO learn creating IOT Applications

• Follow the IOT Application development tutorial from Snap4City
  – 3° Part
  – https://www.snap4city.org/577
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

Snap4City (C), January 2022
Data Ingestion of Time Series via IOT Brokers
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

**Knowledge base**
Semantic reasoners

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

**Indexing and aggregating**
OpenDistro x Elastic search

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

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

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

Snap4City (C), August 2020
• IOT Network concepts, See Course 2020 Part 3
  – IOT Devices, IOT vs Dashboards, etc.
• IOT Broker registration if not performed, See Course 2020 Part 3
  – IOT Brokers can be Internal or External (managed by third party)
The process

- **IOT Device registration** on IOT Directory: concept
  - the IOT Directory performs the registration on KB and NIFI for automated Data Shadow storage

- **HOW TO: register IOT Device(s)** on IOT Directory:
  A. Single IOT Device **Manually from Zero** or by **IOT Device Model** (via user interface IOT Dir)
  B. in **Bulk** by uploading a File with IOT Devices via the IOT Directory user interface + CSV file
  C. Automatically via an **IOT App using special Block/MicroService having** an IOT Device Model
  D. Automatically by activating a **querying the IOT Broker** from the IOT Directory (for External)

---

**Diagram Description**

- **IOT Broker Registration**
- **IOT Device Model Reg.**
- **IOT Device Registration**
- **IOT Orion Broker**
- **IOT Directory**
- **IOT App**
- **IOT Orion Broker**

- **Manual in bulk**
- **Sporadic**
- **Static (1c)**
- **Dynamic**

- **NGSI**
- **Real Time**
- **SURI Link**

- **Knowledge Base**
  - Semantic Reasoners

- **Indexing and Aggregating**
  - NIFI, OpenDistro x ElasticSearch

---

Snap4City (C), January 2022
IOT Directory
IOT Network Manager vs Final User

Network of IOT Brokers

External

Internal

IOT Broker

Registering

Discovering

Knowledge Base, Km4City

Knowledge and Storage Data from the Field and City

IOT Directory

Discovering

ServiceMap

Knowledge Base

Discovering

IOT Network Manager

Register

Dashboard Wizard

My IOT Device

Final user Manager

Register

Prepare Deep Manage

Exploit

IOT Application

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

- IOT Directory is a technology of Snap4City ONLY

- Register IOT Brokers
  - Different kind of Brokers, different kinds of authentications and protocols
  - Registered IOT Orion Brokers can be queried for collecting their managed devices (typically for External IOT Brokers), so that those IOT Devices are registered
  - IOT Brokers/Gateways are registered on NIFI to send messages into the Data Shadow, automatically

- Register IOT Devices: singularly or at groups (in Bulk)
  - Registration can be custom or based on IOT Device Model
  - IOT Edge are registered as IOT Devices as well
  - Registered IOT Devices are saved into local DataBase and into the Knowledge Base

- Provide support for security aspects:
  - Generation of Certificates, Keys, etc., according to the model
  - Collection of keys when IOT devices are on some IOT Gateway or Second Level IOT Broker.

- Manage Ownership and Delegation for
  - IOT Brokers, IOT Devices, IOT Device Values also called Sensors/actuators, IOT Device Models
**IOT Directory Features vs Users Roles (10/21)**

<table>
<thead>
<tr>
<th>Entities</th>
<th>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></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Browse, use</td>
<td>Several Tools</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Delegate</td>
<td>API, ..</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Discovery</td>
<td>KB, API, ..</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>IOT Devices</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Browse, use</td>
<td>Several Tools</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Create, change, delete</td>
<td>API, ..</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Register in Bulk</td>
<td>API, ..</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Delegate, Change Owner</td>
<td>API, ..</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Discovery</td>
<td>KB, API, ..</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>IOT Device Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Browse, Use</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>(Yes)</td>
</tr>
<tr>
<td></td>
<td>Create, change, delete</td>
<td></td>
<td>X</td>
<td>X</td>
<td>(Yes)</td>
</tr>
<tr>
<td></td>
<td>delegate, change ownership</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>IOT Broker</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Browse, use</td>
<td>use</td>
<td>Browser, use</td>
<td>X</td>
<td>Yes (use)</td>
</tr>
<tr>
<td></td>
<td>Register/change/Delete</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Deploy Orion Broker</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Delegate</td>
<td></td>
<td></td>
<td></td>
<td>ToolAdmin</td>
</tr>
<tr>
<td></td>
<td>Periodic Update</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>
Get a New Orion Broker
automated deploy of Orion Brokers
Automated Deploy of Orion Brokers

ToolAdmin user access to the button for the automated Deploy of Orion Brokers. This feature can be provided to large Organizations and on demand to medium and small Organizations.

Snap4City (C), January 2022
Automated Deploy of Orion Brokers

• You can define
  – Name/nickname
  – GPS position
  – Accessible from inside and/or outside

• → Automatically....
  – an Orion Broker is deployed in
  – Container for your organization
  – Proxy are set up
  – Registration on data shadow is established (OpenDistro ES)
  – New IoT Device Model and IoT Devices can be registered, etc. (NIFI)
### Deployed Orion Brokers

<table>
<thead>
<tr>
<th>Broker URL: <a href="https://iot-app.snap4city.org/orionfilter/orion-test1">https://iot-app.snap4city.org/orionfilter/orion-test1</a></th>
<th>Broker Port: 443</th>
</tr>
</thead>
<tbody>
<tr>
<td>Latitude: 43.80381</td>
<td>Longitude: 11.20262</td>
</tr>
<tr>
<td>Login:</td>
<td>Password:</td>
</tr>
<tr>
<td>SHA:</td>
<td>Version: v2</td>
</tr>
<tr>
<td>Orion version: 3.10-next</td>
<td></td>
</tr>
</tbody>
</table>

- **You can:**
  - Delete: undeploy the broker from the cloud
  - Change ownership, delegate access
  - See/View/Edit of config parameters
  - Access: public / private
  - External access test
  - Test on internal registration
  - Update to upgrade the Broker at the last Docker version from FIWARE

The broker answers at the link: http://192.168.1.47:2032/v2/entities
IOT Broker Registration
(for External/Internal Orion Brokers or other Brokers)
Snap4city Data Ingestion Flow Diagram

IOT Broker Registration -> IOT Directory

IOT Orion Broker

Knowledge Base
Semantic Reasoners

NGSI

Indexing and Aggregating
NIFI, OpenDistro x ElasticSearch

Information, File
IOT Apps
Snap4City Tools
IOT Device/Gateway

Sporadic (1c) static
Registration triples

Snap4City (C), January 2022
## IOT Brokers

### Table of IOT Brokers

<table>
<thead>
<tr>
<th>IOT Broker</th>
<th>Access Link</th>
<th>Access Port</th>
<th>Kind</th>
<th>Protocol</th>
<th>Ownership</th>
<th>Organization</th>
<th>Created</th>
<th>Edit</th>
<th>Delete</th>
<th>Go to</th>
<th>Utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>orionRisa-UNIFI</td>
<td>192.168.1.47</td>
<td>8447</td>
<td>internal</td>
<td>ngsi</td>
<td>PRIVATE</td>
<td>Pisa</td>
<td>2020-08-03 12:55:52</td>
<td>Co</td>
<td></td>
<td></td>
<td>TEST VIEW</td>
</tr>
<tr>
<td>orionPontDuCard-UNIFI</td>
<td>192.168.1.47</td>
<td>8454</td>
<td>internal</td>
<td>ngsi</td>
<td>PRIVATE</td>
<td>PontDuCard-Crotone</td>
<td>2021-01-15 11:00:54</td>
<td>Co</td>
<td></td>
<td></td>
<td>TEST VIEW</td>
</tr>
</tbody>
</table>

**Broker URI:** 192.168.1.47 192.168.1.47 8447 8454 43.76919 -45.66819 43.76919 45.66819
**Login:** orion
**Password:** orion
**SHA:** orion

### Additional Information

- **IOT Applications**
- **IOT Directory and Devices**
  - IoT Sensors and Actuators
  - IoT Devices
  - IoT Devices Management

---

**Register new IOT Broker**
**Display new orion broker**
Register a New IOT Broker
Snap4 technology is broker Agnostic. Most of the features are only accessible for FIWARE Orion Brokers

- IOT Brokers
  - You can test, view, and register, and also automatically deploy Orion Brokers
  - are associated with an Organization
    - Each **Organization** has a **Knowledge Base** of reference (KB, ServiceMap)
    - Each **KB** may host multiple **Organizations** and addresses multiple **Geographic areas**
  - can be compliant with
    - **NGSI version**: V1, V2-1, V2, etc...
      - with Snap4City Security or regular NGSI FiWare
    - other protocols as well such as: MQTT, COAP, AMQP, etc.

- can
  - expose different **authentication methods**: K1/K2, Certificate, etc.
  - be accessible from IOT Devices and IOT App in **Cloud only**
  - be accessible from **Internet to post data from outside**, etc.
IOT Orion Broker Network: NGSI V1 and V2

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

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

• IOT Brokers can be networked
  – Services, Service paths: for managing the IOT Broker network
  – Multi-tenant: more than one user/org on the same IOT Broker
IOT Device Model
Snap4city Data Ingestion Flow Diagram

Static or quasi Static Data: POI, etc.

IOT Broker Registration

IOT Device Model Reg.

IOT Device Registration

Real Time data in Pull
Any gateway, server
periodic

IOT Orion Broker

IOT Orion Broker

IOT Directory

Knowledge Base
Semantic Reasoners

Indexing and Aggregating
NIFI, OpenDistro x ElasticSearch

Information, File
IOT Apps
Snap4City Tools
IOT Device/Gateway
IOT Device Data Model (1)

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

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

Add new device

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

Edit Model - ChargingStationModel

<table>
<thead>
<tr>
<th>Charging State Value</th>
<th>Charging State (Charging)</th>
<th>Some Coded Status (Status)</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value Name</td>
<td>Value Type</td>
<td>Value Unit</td>
<td></td>
</tr>
<tr>
<td>Ref refresh rate</td>
<td>900</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Healthiness Criteria</td>
<td>Healthiness Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Station State Value</td>
<td>Station State (Station)</td>
<td>Some Coded Status (Status)</td>
<td>String</td>
</tr>
<tr>
<td>Value Name</td>
<td>Value Type</td>
<td>Value Unit</td>
<td></td>
</tr>
<tr>
<td>Ref refresh rate</td>
<td>900</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Healthiness Criteria</td>
<td>Healthiness Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Date/observed</td>
<td>Date/observed (Date)</td>
<td>Date/observed (Date)</td>
<td></td>
</tr>
<tr>
<td>Value Name</td>
<td>Value Type</td>
<td>Value Unit</td>
<td></td>
</tr>
<tr>
<td>Ref refresh rate</td>
<td>900</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Healthiness Criteria</td>
<td>Healthiness Value</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Where</th>
<th>IOT Device Model</th>
<th>IOT Device</th>
<th>A Temporal Instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>IOT Broker</td>
<td>Broker: 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>
• **ID**: is the unique identifier for reconnecting Temporal Instances with registered IOT Devices

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

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

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

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

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

  – *Your biscuit is not changing if the template is modified after the printout*
IOT Device Registration

many possibilities
Activities for IOT Device Registration

• Registration of
  – N IoT Devices and their IoT Device Model via Data Table Loader
  – an IOT Device Manually from Zero or from an IOT Device Model
  – a set of IOT Devices with the same IOT Device Model from IOT App
  – a set of IOT Devices in BULK loading a CSV (with or without a reference IOT Model)
• The IOT Device registration implies the automated production of the Digital Twin Device into the Knowledge Base, which implies:
  – Activation of the Storage “DataShadow” for historical data access
  – Activation of all the relationships
  – Activation of Discovery mechanisms via IOT Directory, KB
  – Activation of Dashboard Wizard (after a few minutes), and Data Inspector
  – Etc.
IOT Devices/Sensors

Snap4City (C), January 2022
IOT Discovery on IOT Application Development
IOT Device Registration from IOT Device Model
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

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

Addition of Static Attributes of the IOT Device

Only if you enabled from model
IOT Device Registration in Bulk from CSV File

IOT Directory and Devices
- My IOT Sensors and Actuators
- IOT Sensors and Actuators
- IOT Devices
- IOT Devices Management
- IOT Device Discovery
- IOT Brokers
- IOT Device Models
- IOT Devices Bulk Registration
- IOT Broker Periodic Update settings
- IOT Orion Broker Mapping Rules
- Doc: IOT Directory and Devices
- Create an IOT Device Instance
- Create an IOT Device Model
- Add an IOT Device into Snap4City
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
File model in https://www.snap4city.org/289
Get a CSV template of your device for filling it

- download a csv template conformant to the IOT Device Model,
  - Go on the IOT Device Model, click on (+) and press the EXPORT button
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)
Snap4city Data Ingestion Flow Diagram

- **IOT Device Registration**: pull
- **IOT Directory**: (1c) static, registration triples
- **IOT Orion Broker**: dynamic, subscription note, Real Time
- **Knowledge Base Semantic Reasoners**: SURI Link
- **NGSI Indexing and Aggregating**: NIFI, OpenDistro x ElasticSearch

- **CSV**, **XLS**, **JSON**: IOT Device/Gateway
- **IOT Apps**: IOT Orion Broker
- **Snap4City Tools**: NGSI
- **Information, File**: IOT Apps
- **IOT Device/Gateway**: Snap4City Tools

Snap4City (C), January 2022
1) Model creation

Model name: Florence wifi average person

Dataset Creation

Snap4City (C), January 2022
2) IoT Devices Creation from IOT APP

BLOCK: ‘IoTDirectory-new-device-from-model’
Model name: Florence wifi average person
3) Group Creation (more than 200 devices) -> put all the devices in the group and put them as ‘public’ (or they remain private)
4) Send RT data to the IoTDevices
5) Verify RT Data via Snap4City API or via ServiceMap
1) IoTModel

2) Static Flow to create IoTDevices

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

5) Working on Dynamic Flow to save Average #people every 15 minutes for each IoTDevice
Real Time Data Ingestion via IOT Applications

See Training Course Part 3 for Time Series data coming from several kinds of IoT Devices: mobile devices, Libelium, SigFOX, FIWARE, etc.
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

Knowledge base
Semantic reasoners

Search and Query,
Smart City API
Facet, semantic search

Indexing and aggregating
OpenDistro x Elastic search

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

IOT Applications, Business Logic
Node-RED + Snap4City MicroServices

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

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

Back-End

Snap4City (C), August 2020
Snap4city Data Ingestion Flow Diagram

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

**Data Flow:**
- **Push** from **IOT Devices** to **IOT Brokers**
- **Push** from **IOT Brokers** to **IOT App**
- **Push** from **IOT App** to **Adapter**
- **Push** from **Adapter** to **IOT Orion Broker**
- **Push** from **IOT Orion Broker** to **IOT Directory**
- **Pull** from **IOT Directory** to **IOT App**
- **Pull** from **IOT App** to **Adapter**
- **Pull** from **Adapter** to **IOT Orion Broker**
- **Pull** from **IOT Orion Broker** to **Knowledge Base**
- **Registration triples** from **IOT Directory** to **Knowledge Base**

**Types of Data:**
- **Static** (1c), **Dynamic**, **Sporadic**, **Periodic**

**Technologies:**
- **NIFi**, **OpenDistro**, **ElasticSearch**, **Snap4City Tools**
A range of Possibilities

- Any Data can be ingested from any kind of service via an IOT App by using any kind of protocol
- IOT Devices NGSI, MQTT, ... can push data into Internal Brokers Snap4City
- IOT Devices with other protocols can connect with IOT App for adapting their data to the data of the Internal Brokers Snap4City
- IOT Devices can be connected to any IOT Broker of any protocol and the IOT Broker can talk with Internal Brokers Snap4City, with or without adapter

THUS any data passing on our Internal IOT Brokers is automatically indexed into Elastic Search and Knowledge Base creating the full support for query and data shadow, usable immediately without any config for dashboards, synoptics, data inspection, etc.
Dynamic Data

- **Hyp:** IOT Device created

- Create an IOT App to send the dynamic data to your Device/s (in push or Pull)

- The IOT App is based on the data you have:
  - Data ingestion
  - Data transform to a json referring to the IOT Device/s
  - Choose the correct broker

Snap4City (C), January 2022
Dynamic Data

- Charging Station in Florence (Open Data)
  1. Use an **inject** block to chose the frequency of update
  2. Use **http** block to download data
  3. Use a set of blocks to convert data from xml to json
  4. Use a **function** block to create a json referring to your Model and Devices (ChargingStationModel)
  5. Use a **fiware Orion** block to send the dynamic data:
     - selecting the broker
     - Put your K1, K2

ChargingStationModel: values

Snap4City (C), January 2022
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
Further readings

• HOW TO: add a device to the Snap4City Platform
• HOW TO: add data sources to the Snap4City Platform
• HOW TO: add IOT Device data source from external broker to the platform.
• TC9.13: How to upload a local file into your IOT Application
• TC9.2. Managing heterogeneous File Ingestion, protocols, formats via IOT applications, and open standards
• TC2.25. Registering external MicroService calling RestCall services, using it on IOT applications
An Alternative IoT Data ingestion by using a range of IoT Brokers
Registration and Data Driven with IoT Brokers

Case A)

IOT Broker Registration

| sporadic | (1c) static |

Your IoT Broker

IOT Devices

IOT Orion Broker

IOT Orion Broker

IOT Directory

registration triples

NGSI

Knowledge Base
Semantic Reasoners

subscription note

Indexing and Aggregating
NIFI, OpenDistro x ElasticSearch

Real Time

SURI Link

Case B)

IOT Devices

Information, File

IOT Apps

Snap4City Tools

IOT Device/Gateway
Registration and Data Driven with IoT Brokers

Case B)

IOT Broker Registration

Your IoT Broker or Gateway

IOT Devices

Real Time

IOT Orion Broker

NGSI

IOT Directory

registration triples

Knowledge Base
Semantic Reasoners

SURI Link

Indexing and Aggregating
NIFI, OpenDistro x ElasticSearch

Using an Adapter for protocols via Node-RED, IoT App
IDAS as Agent for connecting other protocols

- FIWARE NGSI is capable to deal with the wide variety of IoT protocols today.
- Rather than trying to solve the battle of standards at IoT level, it brings a standard where no standard exists today: context information management.
A Complete Example for Time Series: IOT Device Model + IOT Data Ingestion
I have created an IOT Device Model as:

```
statuscorregione

Name
misura
Device Type
protezione civile
Producer
Healthiness Criteria
Automatically generated
Key Generation

statuscorregione
Description
Sensor
Kind
600
Frequency
Healthiness Value

Edge-Gateway Type

onionUNIFI
ContextBroker
json
Format
ServiceTenant
ServicePath

ngt
Protocol
```

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

- Snap4City engine recognizes as time basis for the TimeSerie only 1 Variable with
  - ValueType as TimeStamp (in milliseconds)
- If you need more than one timestamp in milliseconds use:
  - ValueType as DateTime (in milliseconds)
From IOT Model I have created some instances: the IOT Devices

<table>
<thead>
<tr>
<th>Device Identifier</th>
<th>IOT Broker</th>
<th>Device Type</th>
<th>Model</th>
<th>Ownership</th>
<th>Status</th>
<th>Edit</th>
<th>Delete</th>
<th>Location</th>
<th>View</th>
</tr>
</thead>
<tbody>
<tr>
<td>admnDev1</td>
<td>orionUNIFI</td>
<td>Ambiential</td>
<td>PUBLIC</td>
<td>active</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>alert_161054239306</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td>VIEW</td>
</tr>
<tr>
<td>alert_161054634047</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td>VIEW</td>
</tr>
<tr>
<td>alert_161061389703</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td>VIEW</td>
</tr>
<tr>
<td>alert_161062997473</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td>VIEW</td>
</tr>
<tr>
<td>alert_161077479380</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td>VIEW</td>
</tr>
<tr>
<td>alert_161075905427</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td>VIEW</td>
</tr>
<tr>
<td>alert_161075997465</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td>VIEW</td>
</tr>
<tr>
<td>alert_161077702089</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td>VIEW</td>
</tr>
<tr>
<td>alert_161077124769</td>
<td>orionUNIFI</td>
<td>event</td>
<td>AlertGeneric</td>
<td>MYOWNPRIVATE</td>
<td>active</td>
<td>EDIT</td>
<td>DELETE</td>
<td></td>
<td>VIEW</td>
</tr>
</tbody>
</table>

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

- **NAME** (it has to be unique)
- Select the IOT Model: «statuscorregione»
  - Thus the K1, K2 appears since the model is associated to an Orion Broker that needs to have them, the tool generate them for you but you can impose if you like
  - See in previous slide the ID name of the IOT Broker used
- **Lat** and **Lon**, GPS coordinates you can:
  - pick on the map
  - Write the coordinates manually and see the pin on map
Once Created, I may send a new data to it

Get/See last message from Broker

Generate a New Message towards the Device, Storage

View IoT Device on map and its last value
Impose current date time on dateObserved

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

- You may create an IOT App, where:
  - Function: is preparing the JSON package
  - Block «Fi-Ware Orion OUT V1» 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
NGSI versions

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

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

- Orion Broker of V1 with NGSI syntax of V1

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

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

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

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

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

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

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

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

Alcuni dati in passato non sono pervenuti alla protezione civile

JSON for Authentication as well

```json
msg.auth = {
  "k1": "1ef0e5e8-yyyy-xxxx-9462-0aa4cfef5e19",
  "k2": "b2b34425-yyyy-xxxx-818d-2d6cac2314a6",
  "apikey": "apikey",
  "basicAuth": "basicAuthKey"
};
```
You may use other functions from IOT Directory

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

- You can create smart IOT Applications that on the basis of the list of IOT Devices would request all what you need to load data into YOUR OWN IOT Devices including:
  - Service URI
  - K1, K2
  - Authentication
Real Time Data Ingestion for Industry 4.0 cases
Snap4Altair Decision Support supervision and control, Industry 4.0

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

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

• Historical and Real Time data
  • Billions of Data

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

• Since 2020
Snap4Industry IOT Architecture
## Optimized Production Planner

<table>
<thead>
<tr>
<th>Parameters (TabPar)</th>
<th>DCS (OPC-UA)</th>
<th>Administrative data (AS400)</th>
<th>Administrative Consolidated Planning data (AS400)</th>
<th>Energy data</th>
<th>Other Parameters</th>
<th>Planning result</th>
<th>Outcome</th>
<th>In production</th>
</tr>
</thead>
<tbody>
<tr>
<td>2020-09-25 18:47:36</td>
<td>2020-10-21 18:00:02</td>
<td>2020-10-21 17:59:47</td>
<td>2020-10-21 17:59:47</td>
<td>2020-10-22 23:00:00</td>
<td>2020-07-24 18:43:00</td>
<td>2020-10-21 18:01:12</td>
<td>completato</td>
<td>Sì</td>
</tr>
</tbody>
</table>
Produce Optimized Production Plan

Data Storage
- List of Active Orders
- Consolidated Orders
- Last Params
- Take last DCS
- Take last Energy

Consolidated Orders

Start Planning

Consolidated

Decision Support

Possible Plans

Production Plan

Possible Plan
Some Flows
Data Ingestion via API: External services, using HTTP MicroService on IOT App
General solution, bring data from API to Dashboards

• You can use the MicroServices HTTP in get and post to act on REST Call

• OR

• 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

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

- Nature: Transfer service and renting
- Sub Nature: Monitoring camera
- Licence: Public
- Description: Antwerp cameras location from A Open Data
- Select Image: Nessun file selezionato
- Method: GET
- Do you want create a Microservice with Authentication?:

Help:
- Description of microservice
- The service gives the camera location (lat, lon)
- Inputs
- Microservice input description:
- No Parameter
- Outputs
- json
- Details
- More details here: https://opendata.antwerpen.be/datasets/kaart

Remove Parameter  Add Parameter

Cancel  Confirm
Usage of the MicroService from IOT App
Data Ingestion via IOT App towards MyKPI
Snap4City Architecture vs Data Ingestion

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

Knowledge base
- Semantic reasoners

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

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

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

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

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

Data Sources, External Services
- PULL Data

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

Snap4City (C), August 2020
MyKPI can

- be time series + metadata
- be POI with full metadata decryption, MyPOI
- be passed in ownership to other users,
- be delegated in access to other users
- model daily trajectories from: Mobile Phone Apps, CANBUS data and GPS location from mobiles, PAX Counter Mobile, mobile IOT Devices, etc.
- be saved and retrieved from IOT Apps
- create events at their changes towards IOT Apps
- be saved into: MySQL and/or OpenDistro x ElasticSearch (default saving modality can be different for solutions on cloud and on premise)

• The access from smart city API is in any way transparent, while only when they are saved into the OpenDistro x Elastic Search they are accessible from the so called
  – “My Data Dashboard Kibana”
Create your MyKPIs

- My Data, KPI, POI > ‘Add My KPI’ button
- Verify the KPI existence in My Data, KPI, POI
- Create your IoT App sending data to your KPIs
- Example: Lonato Car Park:
  - NumFreeSlots
  - MaxDuration
  - MaxDurationSlotId
IoT App sending data to your KPIs

- Create your IoT App (ex: ‘SmartParking LonatoDelGarda’)
  1. Use an **inject** block to choose the frequency of update
  2. Use **http** block to download data (e.g.)
  3. Use a **function** of blocks to convert data in a specific json sending data to each KPI
  4. Use a **save-my-kpidata-values** block

Snap4City (C), January 2022
Data Ingestion via IOT App
web Scraping
Snap4City Architecture vs Data Ingestion

Data Sources, External Services
DATA SOURCES, PULL DATA

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

Knowledge base
Semantic reasoners

Search and Query, Smart City API
Facet, semantic search

Indexing and aggregating
OpenDistro x Elastic search

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

IOT Applications, Business Logic
Node-RED + Snap4City MicroServices

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

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

Back-End
Web Scraping from Portia

My Scraping process

Web Scraper PORTIA

Generating WEB Scraping

Sharing/saving reusing Scraping

IOT App. Editor

Resource Manager

Knowledge Base, Km4City

Snap4City (C), January 2022
See them in the list of IOT Apps
Web scraping

- TC9.16 – Web Scraping to get data from web pages
Data Streams from Smart City API, participatory
• We intend in this cases the data that can be posted on the infrastructure by using the Smart City API, such as:
  – MyPersonalData, MyKPI, data from IOT App on mobile, etc.
  – please see slides of Day 7
• Next section discusses those that are automatically collected from Mobile Phone and sent to the infrastructure on cloud via the Smart City API. For example:
  – Clicks on App
  – Post of images on POI
  – Post of comments on POI
  – Post of raking on POI
  – Questionnaires, and reactions
  – Eventual scores collected on the basis of the actions performed by the users.
  – Trajectories taken from the mobile phone positions and clicks
  – OBD2 data and positions
  – Etc.
Data Streams from Mobile Devices
Snap4City Architecture vs Data Ingestion

- **Data Sources, External Services**
  - Data Ingestion, aggregation, regularization, reconcile:
    - PULL Data
  - NIFI, IOT App

- **Knowledge base**
  - Semantic reasoners

- **Indexing and aggregating**
  - OpenDistro x Elastic search

- **Search and Query**
  - Smart City API

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

- **Data Ingestion in Push**
  - Data ingestion
  - Node-RED + Snap4City MicroServices

- **IOT Applications, Business Logic**
  - Inform, announce, Act!, warning, alarms, What-IF, ..
The App is a Bidirectional Device

- GPS Positions
- Selections on menus
- Views of POI
- Access to Dashboards
- searched information
- Routing
- Ranks, votes
- Comments
- Images
- Subscriptions to notifications
- ....

Produced information
- Accepted ?
- Performed ?
- ...

Users

Derived information
- Trajectories
- Hot Places by click and by move
- Origin destination matrices
- Most interested topics
- Most interested POI
- Delegation and relationships
- Accesses to Dashboards
- **Cumulated Scores from Actions**
- Requested information
- Routing performed
- ..... 

Produced information
- Suggestions
- Engagements
- Notifications
- ...

System
Profiled Engagements to City Users

• The users are profiled to learn habits:
  – Personal POI, paths, Mobility habits
• Information and engagements sent to the users are programmed according to the context and user behavior to:
  – Stimulate virtuous habits
  – More sustainable habits
  – More healthy habits, etc.
  – Get feedbacks
  – Provide bonus and prices, ..... 
  – Send alerts, .....
Users’ Engagement

**Inform**
- Air Quality forecast is not very nice
- You have parked out of your residential parking zone
- The Road cleaning is this night
- The waste in S.Andreas Road is full

**Engage**
- Provide a comment, a score, etc.

**Stimulate / recommend**
- Events in the city, services you may be interested, etc..

**Provide Bonus, rewards if needed**
- you get a bonus since you parked here
- We suggest: leave the car out of the city, this bonus can be used to buy a bus ticket

**Rules**

**User context**

**City context**

Snap4City (C), January 2022
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

Apps

Mobile

PAX Counter

OBD2

Mobile

sensors
Load and change data via Smart City API

• Data regarding Mobile Phones:
  – Clicks on App
  – Post of images on POI
  – Post of comments on POI
  – Post of raking on POI
  – Questionnaires, and reactions
  – Trajectories taken from the mobile phone positions and clicks
  – OBD2 data and positions
  – etc.

• They are **automatically collected** and can be inspected by the user via special tools as presented in the following.
My KPI data view and manipulation

https://www.snap4city.org/mypersonaldata/
Further reading on MyKPI

- TC1.17. Object tracking, widget tracker, personal tracking/trajectories, moving sensors
- TC 2.35- How manage My KPI with Dashboard
- Solution: using PAX Counters, monitoring museum and events
- HOW TO: define privacy rules for personal data, produced by the end-users own device
- TC9.13: How to upload a local file into the platform (IOT Application and MyKPI)
Data Streams from Dashboards
Snap4City Architecture vs Data Ingestion

Data Ingestion in Push

Data Sources, External Services
PULL Data

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

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

Knowledge base
Semantic reasoners

Search and Query,
Smart City API
Facet, semantic search

Indexing and aggregating
OpenDistro x Elastic search

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

IOT Applications, Business Logic
Node-RED + Snap4City MicroServices

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

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

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

Data ingestion in Push
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 (C), January 2022
IOT Data Driven

Snap4City Platform storage for «Data Shadow» and much more

Towards any IOT Device and/or Dashboard

Dashboards also provide rendering for sensor values

Managing Public and Private IOT/IOE Devices

Snap4City IOT Brokers

Real Time

Real Time + Historical

Sensor Actuator

From Dashboard to IOT Device

From IOT Device to KB

Actuator

Sensors

IOT Applications

Sensor Actuator

From Dashboard to IOT Device

From IOT Device to KB

Actuator

Sensors

IOT Applications

Snap4city Platform storage for «Data Shadow» and much more

Towards any IOT Device and/or Dashboard

Dashboards also provide rendering for sensor values

Managing Public and Private IOT/IOE Devices

Snap4City IOT Brokers
What the Dashboards can directly exploit

- IOT Device
- IOT Broker
- IOT Edge
- GIS data, Maps,
- External Data Stores
  - LD, LOD
- Snap4City BigData Storage and KB
- External Services
  - API, External Services
- Web Scraping
- DataGate
  - ckan

Dashboards

- IOT Application
- From IOT App to Dashboard
- From Dashboard to IOT App
- Big Data Analytics,
  Artificial Intelligence
  Access to all data

Snap4City (C), January 2022
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)
From Dashboard to IOT App

MyKPI variable onchange

Synoptics

MapClick

By Changing

Numeric keyboard

Switch button

Dimmer

Geolocator

Dropdown

Form

Coordinates from map

Event driven

My KPI

Synoptic read

Synoptic subscribe

Snap4City (C), January 2022
IOT Application

From IOT App to Dashboard

Nature

Snap4City (C), January 2022
<table>
<thead>
<tr>
<th>Widgets ICONS</th>
<th>Widget Name, Description</th>
<th>IOT App</th>
<th>Dashboard-IOT App</th>
<th>KPI (metric)</th>
<th>MyPersonalData</th>
<th>MyData</th>
<th>MyKPI</th>
<th>Sensor</th>
</tr>
</thead>
<tbody>
<tr>
<td>![XX]</td>
<td>Single Content</td>
<td>![single content]</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>50</td>
<td>Speed Limit (see custom for more)</td>
<td>![speedometer]</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>![clock]</td>
<td>Speedometer</td>
<td>![gauge chart]</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Single Bar, V/H</td>
<td>![vertical single bar]</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Single and Multiple Bars, stacked or not, ordered</td>
<td>![Bar series]</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>MultiSeries, shaded, staked and non staked, TTT</td>
<td>![curved line series]</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>![chart]</td>
<td>Time Trend (single)</td>
<td>![time trend]</td>
<td>X</td>
<td>X (DD)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Time Trend Compare</td>
<td>![radar series]</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SpiderNet, radar, Kiviat</td>
<td>![pie chart]</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Pie, Donut, 2 layers Donut</td>
<td>![table content]</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Table</td>
<td>![calendar]</td>
<td>X (cs)</td>
<td>X (DD)</td>
<td>X</td>
<td></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 it stored into Sensor or MYKPI variable, etc.
  – **CS, widget supports Change Source**, in the sense that: from the IOT App it 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 reloading

*TC4.9: New Support Widgets for Bars, Barseries, Trend, and Series, on Dashboards and IOT Applications* (partially obsolete)
From-To Custom Widgets / Synoptics to Storage in WS

MyKPI

Sensors

MyKPI

Sensor

New Shared Variables

Constant Values

Web Socket Secure
GIS Data Import and Export: WFS and WMS

https://www.snap4city.org/drupal/node/368
GIS vs Sna4City

GIS Server can be: ESRI ArcGIS Enterprise, QGIS, GeoServer, ..
GIS Player can be: ESRI ArcGIS Pro, ArcGIS Portal, Snap4City WFS player, ...

• GIS:
  • Geographic Information System
• WMS:
  • Web Map Service
• WFS:
  • Web Feature Services

Any External Service

Dashboard System

GIS Player

GIS
WFS
WMS

GIS
WFS
WMS

Smart City APIs

Data Ingestion

IOT Apps

Big Data and Semantic Storage

WMS/WFS APIs

Big Data Analytics, Artificial Intelligence

IOT & Real Time Streams

All Real Time Streams

Dashboards and Apps

EA

A

B

C

D

E
• Snap4City is interoperable with
  – ESRI ArcGIS Enterprise, Portal, Pro/MAP, ...
  – other GIS tools supporting WFS, WMS, GeoJSON, GML

• Snap4City is interoperable since:
  – Provides info/data in WFS, WMS
  – Exploits data/info from WFS, WMS
  – Import data/info from WFS/WMS

• The Snap4City platform can be installed on premise using Snap4City Appliance
  https://www.snap4city.org/drupal/node/471
  – StartSNAP4CITYVM includes the Dashboard Builder that is capable to work with WFS WMS protocols for the integration with GIS platforms as ESRI ArcGIS, QGIS, directly or using Snap4City GIS player.
  – KBSSMVM includes the Smart City API and WFS API which can be used to data harvest from any GIS servers and GIS desktop tool
ArcGIS ESRI as External Service

- DISIT Lab has ESRI ArcGIS Enterprise 10.6 installed
- ArcGIS Portal accesses to ArcGIS Enterprise server
  - https://arcgis.km4city.org/arcgisportal/apps/Embed/index.html?webmap=5774fa9d0f7f4a2fb2d7cd52d0ffcd8c9&extent=12.3144,45.4276,12.3461,45.4419&home=true&zoom=true&previewImage=false&scale=true&search=true&searchextent=true&details=true&legendlayers=true&active_panel=details&basemap_gallery=true&disable_scroll=true&theme=light
- Snap4City Dashboard uses as External Service: ArcGIS ESRI Portal

Snap4City (C), January 2022
GIS data on Dashboard via Snap4City GIS Player

- DISIT Lab has ESRI ArcGIS Enterprise 10.6 installed
- Snap4City has its WFS Player https://main.snap4city.org/widgets/venezia/index.php
- Snap4City Dashboard uses as External Service: Snap4City GIS viewer via WFS/WMS: https://main.snap4city.org/view/index.php?iddasboard=MTIwNg==

Snap4City (C), January 2022
Dash with Snap4City GIS widget and Selector

- DISIT Lab has ESRI ArcGIS Enterprise 10.6 installed
- Snap4City has its WFS / WMS widget / Player
- Snap4City Dashboard shows WFS/WMS data via Special GIS Widget Map:
  - https://www.snap4city.org/dashboardSmartCity/view/index.php?iddasboard=MTQwMw==
  - Snap4City can use Selector to select WFS / WMS sources to be shown from ESRI ArcGIS (as well as from any other WFS service) on Widget map

The Snap4City Widget Map allows to mix WFS GIS sources with Smart City API
https://www.snap4city.org/dashboardSmartCity/view/index.php?iddasboard=MTM5NA==
Dashboard with Orthomaps and shapes layers
based on WMS, GeoJSON

- DISIT Lab has ESRI ArcGIS Enterprise 10.6 installed, and GeoServer
- Snap4City main MultidataMap Widget can load WMS background images and shapes, the Orthomaps
- Also Heatmaps calibrated are provided from GeoServer using WMS protocol
- Maps in this case are directly taken from ESRI Server, free of charge
- https://www.snap4city.org/dashboardSmartCity/view/index.php?iddasboard=MTQwNg==#
Snap4City via WFS on top of Smart City API provide data to ESRI ArcGIS Enterprise or ArcGIS Pro, and thus the data become accessible on ArcGIS ESRI Portal

- Snap4City provides a WFS service on top of SmartCity API. Thus providing data to any GIS/WFS client, including ESRI ArcGIS Enterprise, ArcGIS Pro, QGIS, etc.
- In the example, our ingested Helsinki Data have been harvested from ESRI ArcGIS via WFS.
- Once ingested on ESRI ArcGIS can be visualized, by using ArcGIS Portal
- Snap4City Dashboard can show ESRI ArcGIS Portal (A) as External Services in a dashboard.
  - https://www.snap4city.org/dashboardSmartCity/view/index.php?iddashboard=MjIwNg==
To test you need to have installed ArcGIS pro on your pc and connect with Snap4City WFS/WMF server or with ArcGIS server which has done the same connection with our server WFS/WMS

On PC:
(i) Get data via WFS connected to ESRI ArcGIS Enterprise or other sources
(ii) Download data via WFS ... from...
(EA) ArcGIS ESRI Pro as GIS / WMS play

To test it you need to have installed ArcGIS pro on your pc and connect with our WFS/WMF server or with ArcGIS server which has done the same connection with our server WFS/WMS.

Heatmap taken from Snap4City GeoServer via WMS protocol.
Snap4City vs GIS, WFS/WMS

• GIS data:
  – Ingested via WFS/WMS protocols, and then managed as the other data. Data ingestion from GIS server can be performed via ETL processes, or directly from Dashboards
  – Shown on Dashboards via third party GIS tools as External Services
  – Shown on Dashboards using Special GIS Widget Map which directly access to GIS data via WFS/WMS
  – Heatmaps and Maps are distributed via a GeoServer

• Snap4City can interact with ArcGIS Real Time Events via MQTT protocol as well

• Snap4City vs GIS solutions and connections
Integration with CKAN
Open Data Manager and Portal
Automatize:
• Import data from CKAN to Snap4City
• Upload Public Data from Snap4City to CKAN
• Data Harvesting
• Dashboards and Mobile/Web Apps creation

Snap4City vs CKAN

Advanced Snap4City APIs and Micro Services

Datagate

Harvesting and Publishing

Open or Private External CKAN Data Portals
Almost all the calls to CKAN are quite similar
Read more on

- **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**
Integration with Copernicus Satellite Data
In the Smart City context there is the needs of

- **Accessible and affordable** data: spatially and temporally dense
- **Reducing costs** for data gathering.
  - Sensors are good, but are scattered and very expensive
  - Reduce **costs for maintenance** of data gathering solutions
    - Sensors have high costs of maintenance: repairing, battery changes, calibrations, attacks, etc.
- **Validation** of data.

- Satellite data may be a solution to some of those problems, while other have to be managed.
Smart City: Satellite Data vs Sensors Data

- **From Satellites, many sources, different resolutions, open/closed:**
  - Ozone, NO2, SO2, Aerosol, CO, etc.
  - Temperature, vegetation, land usage
    - Evolution of soil usage: with high seasonality, and weather impact
  - Air traffic derived data
  - Water traffic usage data
  - Many other technical measures....
    - Spatial and temporal resolution ???

- **From Sensors and other sources:**
  - Pollutant: PM10, PM2.5, NO2, NO, SO2, CO2, ...
  - Weather: temperatures, humidity, wind, DEW, etc.
  - Other: Traffic flow sensors, people flow, parking, etc.
  - Air/lidar measures from flights: vegetation, land usage
  - Scattered data, specifically positioned, no dense data
Satellite data

• A large number of measures, not accessible from ground level sensors

• Complex data stream acquisition
  – Data Transformation by knowing the satellite model is needed
  – Complex for small area since satellite products are typically large area

• Temporal and spatial resolutions (lat, lon)
  – They are not matrices actually
  – They are not always taken on the same places
  – Resolution may be not enough for specific city area analysis
  – No event driven data

• View from the space:
  – Affected by cloud and weather
  – Measures of the column of air and not at the ground level
Sentinel-3 reference geocentric latitude and longitude, time step of 1 second
Example of Copernicus Data

Air Quality Copernicus

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

Snap4City (C), January 2022
Satellite Data Harvesing and Preparation

1. **GET token**

2. Insert metadata (unprocessed data)

3. **GET**

4. **GET** metadata

5. **Download data**

6. Processed data

7. Get metadata

8. **Heatmap production**

9. Get heatmap metadata

10. GeoTIFF

Snap4City (C), January 2022
Compernicus Data Request: Sci-Hub

https://www.snap4city.org/671

Map name: ................

Metric name: AirTemperature, Humidity, Altitude, OLCI Global Vegetation Index, Cloud Fraction, etc.

Description: a generic description;

Location: select the level the data have to be taken and heatmap created. It is possible to specify one of the following: City, Country, State or Postal Code;

Location Name: specify here the location: the name of a City or "Città Metropolitana di Firenze", or "Toscana" as State or "Italy" as Country, etc.;

Color Map: color map visualization for example: airHumidity, ogvi, altitudeHQ, airTemperatureHQ, FractionalCloudCoverLQ, ... From those of Snap4City

Org: specify the organization in Snap4City from the available list;

From Date - To Date: use these to forms to specify the time period of the data to be downloaded. Please note that at least you have to specify at least 1 day period since satellite data are typically updated 1 times per day. If a longer period is specified, all data included in the period will be taken and, according to the available data, more date sets and heatmaps will be generated covering the time period;

Length: specify here the dimension in meters of squared area, for example 700 for obtaining points values in a grid of 700x700 meters;

Write: (1) to have data on piking and database, or (0) to do not have data thus obtaining only the heatmap;

You need to have a TOKEN to use the service 😊
Copernicus data request via IoT Apps

```json
msg.payload = {
    "map_name": "AirTemperatureBolognaCopernicus",
    "description": "Air Temperature Bologna",
    "location": "city",
    "location_name": "Città metropolitana di Bologna",
    "color_map": "airTemperatureHQ",
    "org": "DISIT",
    "from_date": "2021-01-01T00:00:00",
    "to_date": "2021-01-01T23:59:00",
    "length": 700,
    "write": 1,
} return msg;
```
Once Generated can be exploited

- Picking data on dense map and exploiting them on
  - Assessing routing:
    - path of GPS points
  - Alerting specific users wrt specific locations.
    - One GPS position: park, garden, house, etc.
    - Alerting them
      - Via telegram
      - Email

- Estimating city Indexes

- Comparison with sensors
Heatmap production and used satellite and IOT data integrated

- Satellite Data Services
- Computation of Data Value on Grid
- Scattered IOT Sensor Data of Value \{V\}
- Data GRID \{V'\} of \{V\}
- HeatMap Data Server
- HeatMap GeoTIF Generation
- Satellite Data Harvester
- Order Interface
- Satellite Order data
- GeoTIF of \{V'\}
- GeoServer
- Points
- Alerting Assessment
- Condition ed Routing
- Comparison
- FromTo
- Paths
- Alerts
- Point
- Area
- WMS heatmaps
- V’ on Point
- WSs, HTTPS
- Web and Mobile Apps
- MicroServices
- Satellite Data Services
- Computation of Data Value on Grid
- Scattered IOT Sensor Data of Value \{V\}
- Data GRID \{V'\} of \{V\}
- HeatMap Data Server
- HeatMap GeoTIF Generation
- Satellite Data Harvester
- Order Interface
- Satellite Order data
- GeoTIF of \{V'\}
- GeoServer
- Points
- Alerting Assessment
- Condition ed Routing
- Comparison
- FromTo
- Paths
- Alerts
- Point
- Area
- WMS heatmaps
- V’ on Point
- WSs, HTTPS
- Web and Mobile Apps
- MicroServices

Snap4City (C), January 2022

396
15MinCityIndex

FLORENCE metro city

Bologna metro city
Satellite (Copernicus) vs IOT Data

https://www.snap4city.org/dashboardSmartCity/view/index.php?iddasboard=MzAwNQ==
• IoT data have relevant costs for installation and maintenance.

• The exploitation of satellite data in the context of Smart City.
  – PROS: satellite data of the European Union’s Earth observation program Copernicus can be used to
    • calibrate the values of large sensors network data and for new applications in which similar data cannot be recovered from the field.
    • develop new applications in which similar data cannot be recovered from the field.
  – CONS, satellite data are not easy to be managed
    • volume of the data obtained when requesting small regions;
    • complexity of the formats that need to be processed and converted;
    • computational time needed and difficulty in providing data in real time;
    • lack of spatial resolution in providing the data.

• Tools and Dashboards have been provided and integrated into Snap4City suite to:
  – perform a comparison of data coming from satellite with respect to those obtained from IoT devices.
  – Demonstrate that it is possible to create a real time solution by exploiting satellite data
Social Media data collection and exploitation
• Data from Social Media can be ingested in different manners:
  • Twitter Data can be:
    – Collected via Twitter Vigilance programming keywords.
    – Collected in real time via IOT Applications
    – Once collected the Metrics (number of Tweets, reTweets, NLP, Sentiment Analysis, etc. can be:
      • Shown, exploited in IOT Applications
      • Exploited for early warning
  • FaceBook data:
    – Can be generated by using IOT App
  • Telegram data:
    – Can be generated by using IOT App
 Prediction/Assessment

- Football game results as related to the volume of Tweets
- Number of votes on political elections, via sentiment analysis, SA
- Size and inception of contagious diseases
- Marketability of consumer goods
- Public health seasonal flu
- Box-office revenues for movies
- Places to be visited, most visited
- Number of people in locations like airports
- Audience of TV programmes, political TV shows
- Weather forecast information
- Appreciation of services
Twitter Vigilance

- http://www.disit.org/tv
- http://www.disit.org/rttv
- Citizens as sensors to
  - Assess sentiment on services, events, ...
  - Response of consumers wrt, ...
  - Early detection of critical conditions
  - Information channel
  - Opinion leaders
  - Communities
  - Formation
  - Predicting volume of visitors for tuning the services

Snap4City (C), January 2022
Twitter Vigilance RT: sentiment analysis

Real time
Early Warning
Sentiment Analysis
Twitter Vigilance

Early Warning

Predictive models

Attendance at long lasting events: EXPO2015

Attendance at recurrent events: TV, football
IOT and data World

IOT Applications

Dashboards and Apps

My IOT Devices

Big Data Analytics, Artificial Intelligence
Overview (paolo)

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

Snap4City (C), January 2022
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
Data Ingestion and Transformation via ETL Processes

(only for former versions of Snap4City)
Snap4City Architecture vs Data Ingestion

Data Ingestion in PULL

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

- Big Data Cluster (HDFS, HBase, Phoenix)
- Knowledge base
- Semantic reasoners
- NIFI - Tools, ETL, DISCES, - IOT Apps
- Indexing and aggregating (OpenDistro x Elasticsearch)
- Data Analytics (R, Tensor Flow, Python, MapReduce, ...)
- Visual Analytics
- IOT Applications (Node-RED + Snap4City MicroServices)

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

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

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

In Yellow alternative & legacy solutions

Snap4City (C), July 2020
Classic Data warehouse Architecture

Alternative & legacy Snap4City V1
Data Ingestion Flow Guideline, thumb rules

Road Graph Setup

X Data

Static file

Into Data Gate

Automated Execution ETL

Knowledge base

Semantic reasoners

Regular file

ETL: Only PULL

Regularize

Development on Penthao/Spoon

(1) static

ETL Execution on DISCES

Registration

Big Data Cluster

HDFS, Hbase, Phoenix

And/OR

Indexing and Aggregating

NIFI, OpenDistro x ElasticSearch

(2) Real time

ETL Execution on DISCES

periodic

Alternative & legacy Snap4City V1

Snap4City (C), January 2022
Developers of ETL, Data Manager

ETL Process

DISCES scheduler production

ETL Development Environment

Resource Manager

Data Gate

Data Set

Saving / Sharing

Knowledge Base, Km4City

Knowledge and Storage Data from the Field and City

Load data or prepare for data ingestion

Schedule

Execute

Data sources

Alternative & legacy Snap4City V1

Snap4City (C), January 2022
Batch Processing for dynamic data ingestion

Integrated ETL development

Alternative & legacy Snap4City V1
ETL Processes

The three phases are:

• **Extracting** data from outside sources (**Ingestion** phase).

• **Transforming** data to fit operational needs which may include improvements of quality levels (**Data Quality Improvement** phase).

• **Loading** data into the end target (database, operational data store, data warehouse, data mart, etc.). So the data can be translated in **RDF triples using a specific ontology** (Static/periodic datasets) or on **NoSQL Databases** (Dynamic datasets).
Phase I: Data Ingestion

- Purpose is to store data in HBase (Big Data NoSQL database).
- Acquisition of wide range of OD/PD: open and private data, static, quasi static and/or dynamic real time data.
- Static and semi-static data include: points of interests, geo-referenced services, maps, accidents statistics, etc.
  - files in several formats (SHP, KML, CVS, ZIP, XML, JSON, etc.)
- Dynamic data mainly data coming from sensors
  - parking, weather conditions, pollution measures, bus position, ...
  - using Web Services
- Using Pentaho - Kettle for data integration (Open Source tool)
  - using specific ETL Kettle transformation processes (one or more for each data source)
Phase II: Data Quality Improvement

• **Purpose:** add more information as possible and normalize data from ingestion

• **Problems kinds:**
  – Inconsistencies, incompleteness, typos, lack of standards, multiple standards, ...

• **Problems on:**
  – Place-name code
  – Street names (e.g., dividing names from numbers, normalize when possible)
  – Dates and Time: normalizing
  – Telephone numbers: normalizing
  – Web links and emails: normalizing
Phase III: Data mapping

• Purpose is to translate data from QI in RDF triples
• We use triples to do inference on data.

• Using Karma Data Integration tool, a mapping model from SQL to RDF on the basis of the ontology was created.
  – Data to be mapped first temporary passed from HBase to MySQL and then mapped using Karma (in batch mode)

• The mapped data in triples have to be uploaded (and indexed) to the RDF Store (Virtuoso).
• Triples are composed by a subject, a predicate and an object.
Pentaho Data Integration (Kettle)

Main strengths:

• Collect data from a **variety of sources** (extraction);
• Move and modify data (transport and transform) while cleansing, denormalizing, aggregating and enriching it in the process;
• Frequently (daily) store data (loading) in the final target destination, usually a **large dimensionally modeled database** (or **data warehouse**).

• **Spoon**: graphically oriented end-user tool to model the **flow of data** from input through transformation to output (**transformation**)
• **Pan** is a **command line tool** that executes transformations modeled with Spoon
• **Chef**: a graphically oriented **end-user tool** used to model **jobs** (transformations, FTP downloads etc. placed in a flow of control)
• **Kitchen** is a **command line tool** to execute jobs created with Chef.
Type of Steps in Spoon

Three different kinds of steps:

• **Input:** process some kind of 'raw' resource (file, database query or system variables) and create an output stream of records from it.

• **Output:** (the reverse of input steps): accept records, and store them in some external resource (file, database table, etc.).

• **Transforming:** process input streams and perform particular actions on them (adding new fields/new records); these actions produce one or more output streams.
### Type of Transformations in Spoon

#### Transform
- Add a checksum
- Add constants
- Add sequence
- Add value fields changing
- Add XML
- Calculator
- Closure Generator
- Example plugin
- Number range
- Replace in string
- Row denormaliser
- Row flatten
- Row Normaliser
- Select values
- Sort rows
- Split field to rows
- Split Fields
- String operations
- Strings cut
- Unique rows
- Unique rows (HashSet)
- Value Mapper
- XSL Transformation

#### Lookup
- Call DB Procedure
- Check if a column exists
- Check if file is locked
- Check if webservice is available
- Database join
- Database lookup
- Dynamic SQL row
- File exists
- Fuzzy match
- HTTP client
- HTTP Post
- Stream lookup
- Table exists
- Web services lookup

#### Output
- Access Output
- Delete
- Excel Output
- Insert / Update
- Json output
- LDAP Output
- Palo Cells Output
- Palo Dimension Output
- Properties Output
- Salesforce Delete
- Salesforce Insert
- Salesforce Update
- Salesforce Upsert
- Serialize to file
- SQL File Output
- Synchronize after merge
- Table output
- Text file output
- Update
- XML Output

#### Scripting
- Execute row SQL script
- Execute SQL script
- Formula
- Modified Java Script Value
- Regex Evaluation
- User Defined Java Class
- User Defined Java Expression

---

*Snap4City (C), January 2022*
Sequential vs parallel

Start: Get Data from CSV File → Phones → Addresses → Websites → Put Data to Database → Success
• ‘HTTP’ step downloads the dataset defined in PARAM (URL) field.
• If a file with the same name already exists, we compare them.
  - If they are the same, we delete the folders created before.
  - Otherwise we unzip the file and ‘HBase_firstInsert’ is called.
• ‘Get KML’ loads the file just downloaded.
• ‘Get data from XML’ and ‘Row denormaliser’ extract fields from source file.
• In the JS step, we create an identifier (it will be use as key in HBase).
• ‘HBase Output’ saves the information in a HBase table.
• ‘Insert/Update’ updates the last ingestion date in MySQL table process_manager2.
Quality Improvement

- ‘HBase Input’ gets back data saved at the ingestion end.
- ‘Fix address’ is used to correct typing error (e.g. Giovambattista instead of Giovanbattista) or to simplify search of right toponym code.
- ‘Modify_*’ transformations normalize address, CAP, website, e-mail, phone number, ...
- ‘Add constants’ adds two fields (address_syn, codice_toponimo) which we will use in the next job.
- ‘HBase Output’ saves in a new HBase table the quality improvement result.
- For the rows which have an un-empty streetAddress, the steps below extract a word from streetAddress which we will use to find the right toponym code.
To get toponym codes we use *tbl_toponimo_bis* MySQL table. It contains road names, toponym codes, town codes, .... Each road has his center coordinates (lat, long).

- **Table input** does the query on that table using the word created before.
- **Fuzzy match** calculates similarity (from 0.2 up to 1) between the query result and the address acquired during ingestion phase.
- Following steps calculate distance from ingestion coordinates and query ones and select the closest.
- **HBase Output** fills *address_syn* and *codice_toponimo* in QI HBase table.
• The method used for toponym code extraction sometimes produces wrong results.

• Given a word, might be impossible to determine right toponym code (e.g., query using ‘Brunelleschi’ returns ‘Via dei Brunelleschi’, ‘Piazza Brunelleschi’).

• Summarize roads with their centers and calculate distances could not return right results (if the point of interest is far from his road center, it could be closer to another road center).
Since we use Karma to generate models, we have to move data from HBase to MySQL.

- ‘DROP’ and ‘CREATE’ respectively deletes MySQL table if already exists and creates a new one.
- ‘getTime’ returns last triple generation timestamp for a specific process (using MySQL process_manager2 table).
- ‘HBaseToMySQL’ moves data from HBase to MySQL only if timestamp just calculated is older than the date of last ingestion (it prevents to generate triples based on the same data).
- ‘create RegioneCSVTriples’ calls Karma script to generate triples based on the model.
- ‘update last_triples’ updates last triple generation timestamp in process_manager2.
Scheduling Real Time Ingestions

• For **Real Time data** (car parks, road sensors, etc.), the ingestion and triple generation processes should be performed periodically (no for **static data**).

• A scheduler is used to manage the periodic execution of ingestion and triple generation processes;
  – this tool throws the processes with a predefined interval determined in phase of configuration.
## Scheduler: DISCES

### Smart Cloud Engine

**DISIT - Distributed Systems and Internet Technology Lab**

<table>
<thead>
<tr>
<th>SCHEDULER NAME</th>
<th>ID</th>
<th>FIRE INSTANCE ID</th>
<th>DATE</th>
<th>JOB NAME</th>
<th>JOB GROUP</th>
<th>JOB DATA</th>
<th>STATUS</th>
<th>PROGRESS</th>
<th>TRIGGER NAME</th>
<th>TRIGGER GROUP</th>
<th>PREV FIRE TIME</th>
<th>NEXT FIRE TIME</th>
</tr>
</thead>
</table>
Triplification Overview

START

DROP agency

CREATE

getTime

Hbase toSQL

Success

dataset_version, filesytem

Make Triples

Set variables

Evaluate rows number in a table

Delete folders 2

Check if a folder is empty 2

Check if a folder is empty

Delete folders 2

Check if a folder is empty 3

Snap4City (C), January 2022
Triplification Overview

java -cp target/karma-offline-0.0.1-SNAPSHOT-shaded.jar edu.isi.karma.rdf.OfflineRdfGenerator
   --sourcetype DB
   --modelfilepath "${MODELPATH}/${processName}_agency.ttl"
   --outputfile ${TRIPLESDESTDIRECTORY}/agency.n3
   --dbtype MySQL
   --hostname ${IPADDRESSMASTER}
   --username ${USERNAMEMYSQL}
   --password ${PSWMYSQL}
   --portnumber ${PORTMYSQL}
   --dbname ${DATABASEMYSQL}
   --tablename ${processName}_agency

Snap4City (C), January 2022
Karma
An introductory guide for generating RDF triples from relational data in minutes

Karma (http://usc-is-i2.github.io/karma/) is a data integration tool developed at the Center of Knowledge Graphs of the Information Science Institute at the University of Southern California. Here at DISIT, we exploit Karma for triplifying relational data.
• Karma is a mapping model based on ontology (km4city) from MySQL tables to RDF.
• Triples are uploaded to Virtuoso, an RDF Store.
• It can import MySQL tables but no HBase ones.

Here are the steps to have your triples in your pocket in minutes:
1. Get one of the ready-to-use DISIT VMs
2. Launch the Karma server
3. Build your own model:
   1. Load vocabularies
   2. Load relational tables
   3. (Optional) Load R2RML models
4. Define mappings
5. Export your model
6. Launch the command-line tool
7. Enjoy!
Get one of the ready-to-use DISIT VMs

The DISIT Lab makes available through its Drupal portal a set of ready-to-use virtual machines specifically oriented to data integration. Here is how you can get one of them:

1. Connect to [http://www.disit.org/drupal/?q=node/6690](http://www.disit.org/drupal/?q=node/6690) and scroll down to the section “MACCHINA VIRTUALE, VMSDETL, GIA' PRONTA”

2. Get the “Versione del 2017/2018 0.8 con Phoenix” at [http://www.disit.org/vmsdetl/VMSDETL-2017-v0-8.rar](http://www.disit.org/vmsdetl/VMSDETL-2017-v0-8.rar), or the “Versione del 2017/2018 0.8 con Phoenix per Virtualbox” at [http://www.disit.org/vmsdetl/VMSDETL-2017-v0-8-ovf.rar](http://www.disit.org/vmsdetl/VMSDETL-2017-v0-8-ovf.rar), unless you have a good reason for picking a different one

3. Wait for the download to complete, and extract the archive

4. Launch the VM player of your choice

5. Open the VM

6. Run it


Do the following to run the Karma server:
1. Open a shell
2. Move to ~/programs/Web-Karma-master/karma-web
3. Run mvn -Djetty.port=9999 jetty:run
4. Wait while the Jetty server comes up
5. Connect to localhost:9999 where you will find the Web application for building your model
For that Karma could produce the RDF triples for you, it is required that you instruct it about how relational data should be mapped to semantic data. Documents that describe such a mapping are called R2RML models. Models are built operating a dedicated Karma Web application, and they are exported as ttl files.

Steps:
1. Load Vocabularies
2. Load Relational Tables
3. Load R2RML Models
4. Define Mappings
Have you connected to http://localhost:9999? Are you displaying something similar to this? Right, you are ready to load your vocabularies.

Identify classes and properties that you wish to appear in RDF triples that will be the result of the whole process. Identify vocabularies where such classes and properties are defined. Load them.

Below here is how you load a vocabulary:

1. Hit Import, at the top left corner of the Web page
2. Hit From File
3. Select the vocabulary file (it can be an OWL, RDF/XML, or TTL file)
4. Leave OWL Ontology selected, and hit Next
5. Indicate the correct file encoding if the proposed one is not, and hit Import

You should now see your newly imported vocabulary displayed in the Command History (left column).
Identify tables in your RDB where source data can be found. Load them in your model.

Below here is how you load a table:

1. Hit **Import**, at the top left corner of the Web page
2. Hit **Database Table**. The **Import Database Table** dialog should open.
3. Fill in the form with authentication data and RDB name, and hit **OK**
4. A table listing should appear below the form
5. Put the mouse pointer over the table of your interest
6. **Buttons Import and Preview** should appear at the right of the table name. Hit **Import**.
7. Confirmation message “Table imported in the workspace!” should appear. Hit **OK**.
8. Repeat steps 5 – 8 for each table where source data are found
9. Hit **Close** at the bottom right corner of the **Import Database Table** dialog to dismiss it
If you already have built and exported a model in the past, and you now just need to make a modification over it, you can start loading and applying your existing model, instead of rebuilding it from scratch.

Below here is how you apply an existing model:

1. Identify the table to which the model has to be applied, and hit the triangle that is displayed next to the table name.
2. Select Apply R2RML Model, and then From File.
3. Select the ttl file that contains your model, and hit Open.
4. Done. Classes and links should appear in the workspace.
Below here is how you specify that a column of an RDB table maps to a data property of a semantic class, and how to specify that a column contains an identifier that can be used for building the URIs of instances of the semantic class:

1. Below RDB table name, identify the blue box that contains the column name written in white
2. Hit the white triangle that you can see next to the column name
3. Hit Set Semantic Type
4. Pick the checkbox at left of property of Class
5. Hit the Edit button at right of property of Class
6. Select the semantic class from the All Classes list. Use the Class textbox for filtering.
7. Select the property from the All Properties list. Use the Property textbox for filtering.
8. If the column is a key, pick the Mark as key for the class checkbox
9. You can map the column to a typed literal, filling the textbox below the label Literal type
10. When you are done, hit Save. Repeat the procedure for each of the columns to be mapped.
Define Mappings

Below here is how you specify instead that a foreign key of a table in a relational database corresponds to an object property of a semantic class. Scenario: a relational table `stops`, that corresponds to a semantic class `Stop`, has a column `agency_id` where the unique identifier of the agency that manages the stop can be found. Each value in `agency_id` corresponds to one and only one value in a column, let’s say `id`, that can be found in the relational table `agencies`. Table agencies corresponds to the semantic class `Agency`. We wish resources of class `Stop` to be linked each to the appropriate resource of class `Agency`, through the property `gtfs:agency`. For such a purpose, we will do the following:

1. Load relational table `stops` to workspace.
2. Map data properties, linking columns in table `stops`, to the class `Stop`, through appropriate properties, as outlined above.
3. Identify the grey box with rounded angles that has the name of the class `Stop` written within. It should locate in the workspace. Identify the black triangle that should locate near the right margin of the box. Click it.
4. Select Add Outgoing Link. A popup window should open.
5. Type `gtfs:agency` in the box labelled Property, and `gtfs:Agency` in the box labelled To Class.
6. Click Save in the bottom right corner of the popup window to dismiss it.
7. A new grey box, related to class `Agency`, will appear in the workspace, linked to the grey box related to class `Stop` through a link labelled `agency` for brevity.
8. Map column `agency_id` as a data property of class `Agency`, also specifying that it is a unique identifier, as described in the above paragraph.
9. Repeat for all foreign keys to be mapped, then go to next step (Export your model).
Once you have defined all needed mappings, you have to export your model to a ttl file, so that you can provide it as a parameter to the command-line Karma tool that performs the triplification. Here is how you can export your model:

1. Identify the RDB table whose model you wish to export
2. Hit the black triangle at the right of the table name
3. Select Publish, and then Model
4. A popup should appear at the top right corner of the window, saying “R2RML Model published”
5. Hit Manage Models, in the menu bar at the top of the page
6. A listing should appear of all models that you have exported in the current session
7. Identify the row corresponding to the last model exported, based on the File Name (the name of the RDB table) and the Publish Time.
8. Cut the URL that you can find in the rightmost column of the prospect, and open it in a new tab
9. Save As... the page that you have opened at step 8.
10. Done. The file that you have saved at step 9 is your ready-to-use R2RML model.
Launch the command-line tool

Once you have exported your R2RML model as a **ttl file**, you are ready to perform the triplification:

1. Open a shell
2. Move to `/home/ubuntu/programs/Web-Karma-master/karma-offline`
3. Launch the following as a single line command, customizing parameter values in bold:

   ```
   mvn exec:java
   -Dexec.mainClass="edu.isi.karma.rdf.OfflineRdfGenerator"
   -Dexec.args=" --sourcetype DB --modelfilepath /path/to/model.ttl
   --outputfile /path/to/outputTriplesFile.n3 --dbtype MySQL
   --hostname mysql_srv_hostname_or_ip_address --username mysql_user
   --password mysql_pwd --portnumber 3306 --dbname mysql_dbname
   --tablename mysql_table_name -Dexec.classpathScope=compile"
   ```
ETL SDK Virtual machine

- [link](https://www.snap4city.org/download/video/Snap4city_VM_Quick_guide.pdf)
- [videos](https://www.snap4city.org/drupal/node/139)
- example on Github: [link](https://github.com/disit/smart-city-etl)
- [link](https://www.snap4city.org/download/video/Snap4City-ETL-VM.rar)
- “Versione del 2017/2018 0.8 con Phoenix” at [link](http://www.disit.org/vmsdetl/VMSDETL-2017-v0-8.rar), or the “Versione del 2017/2018 0.8 con Phoenix per Virtualbox” at [link](http://www.disit.org/vmsdetl/VMSDETL-2017-v0-8-ovf.rar), unless you have a good reason for picking a different one
Further readings on ETL

- ETL Development: [https://www.snap4city.org/drupal/node/24](https://www.snap4city.org/drupal/node/24)
- US6. Developing and using processes for data transformation
- HOW TO: add a device to the Snap4City Platform
- HOW TO: add data sources to the Snap4City Platform
- TC6.3. Creating ETL processes for automated data ingestion and data transformation
- TC6.4- Managing ETL processes via Resource Manager, upload, execute, monitor
- TC6.5- Managing Heterogeneous File Ingestion via ETL processes
- TC6.6- Producing data-sets in Bundle via ETL
- TC6.8- ETL processes for data transformation, and exploiting MicroServices/API/RestCall
- TC6.9- ETL processes for multiprotocol and format data ingestion, see on GITHUB for library
- TC6.10- ETL Applications using multiple protocols, and formats for files and to calling services using REST and WS
- TC6.11- Add a new ETL coping with a new Protocol
- ETL processes for massive Data Ingestion and Transformation
Acknowledgements
2020
- Smart Tourism
- 6 Pilots
- Data Analytics
- Extended platform

PC4City (2020-21)
Monitoring Terrain

Km4City 1.6.7
- Smart Mobility
- PISA, PUMS
- Living lab

2021

CAPELON
- Smart Light
- Sweden

Smart Ambulance (2021-22)

AMPERE (2021-22)
Industry 4.0

SYN-RG-AI
SmartCity

2022

Winner of Open Data Challenge of PC4City (2020-21)

Almafluida
Industry 4.0 (2021-22)

Enterprise (2021-22)
Industry 4.0

PISA, PUMS
Living lab

PILON

UNI.SYSTEMS
SmartCity

Winner of Open Data Challenge of PC4City (2020-21)

Monitoring Terrain

Km4City 1.6.7
- Smart Mobility
- PISA, PUMS
- Living lab

2022

AMPERE (2021-22)
Industry 4.0

SYN-RG-AI
SmartCity

WINNER OF OPEN DATA CHALLENGE OF PC4CITY (2020-21)

Monitoring Terrain

Km4City 1.6.7
- Smart Mobility
- PISA, PUMS
- Living lab

2022

AMPERE (2021-22)
Industry 4.0

SYN-RG-AI
SmartCity

WINNER OF OPEN DATA CHALLENGE OF PC4CITY (2020-21)

Monitoring Terrain

Km4City 1.6.7
- Smart Mobility
- PISA, PUMS
- Living lab
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)
Overview

Snap4City Platform

Technical Overview

From: DINFD dept of University of Florence, with its DISIT Lab, https://www.disit.org with its Snap4City solution

Snap4City:
- Web page: https://www.snap4city.org
- https://twitter.com/snap4city
- https://www.facebook.com/snap4city

Contact Person: Paolo Neri, Paolo.neri@unifi.it
- Phone: +39-335-5668674
- LinkedIn: https://www.linkedin.com/in/paolo-neri-8499a51/
- Twitter: https://twitter.com/paolone
- FaceBook: https://www.facebook.com/paolo.neri2

Access Level: Public.
Date: 05-04-2021
Version: 5.3

Main running instances (11/21)

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

- Thanks to the European Commission for founding. All slides reporting logo of Snap4City [https://www.snap4city.org](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.
On Line Training Material (free of charge)

<table>
<thead>
<tr>
<th>what</th>
<th>1st part (*)</th>
<th>2nd part (*)</th>
<th>3rd part (*)</th>
<th>4th part (*)</th>
<th>5th part (*)</th>
<th>6th part (*)</th>
<th>7th part (*)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDF</td>
<td>General</td>
<td>Dashboards</td>
<td>IOT App, IOT Network</td>
<td>Data Analytics</td>
<td>Data Ingestion processes</td>
<td>System and Deploy Install</td>
<td>Smart City API: Web &amp; Mob. App</td>
</tr>
<tr>
<td>Interactive</td>
<td>Video1</td>
<td>Video2</td>
<td>Video3</td>
<td>Video4</td>
<td>none</td>
<td>none</td>
<td>none</td>
</tr>
<tr>
<td>duration</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>2:35</td>
</tr>
</tbody>
</table>

https://www.snap4city.org/577
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