



Be smart in a SNAP!

Advanced Smart City API, ASCAPI
Web and Mobile App development

16 January 2020, Course
<https://www.snap4city.org/577>

SCALABLE SMART ANALYTIC APPLICATION BUILDER FOR SENTIENT CITIES



UNIVERSITÀ
DEGLI STUDI
FIRENZE

DINFO
DIPARTIMENTO DI
INFORMATICA
E DELL'INFORMAZIONE

DISIT
DISTRIBUTED SYSTEMS
AND INFRASTRUCTURE
TECHNOLOGIES LAB



UNIVERSITÀ
DEGLI STUDI
FIRENZE

DINFO
DIPARTIMENTO DI
INGEGNERIA
DELL'INFORMAZIONE

DISIT
DISTRIBUTED SYSTEMS
AND INTERNET
TECHNOLOGIES LAB



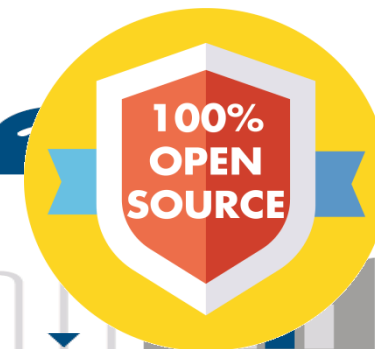
SNAP4city



Powered by

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

<https://www.Snap4City.org>



16 January 2020, Course

<https://www.snap4city.org/577>

Paolo Nesi, paolo.nesi@unifi.it

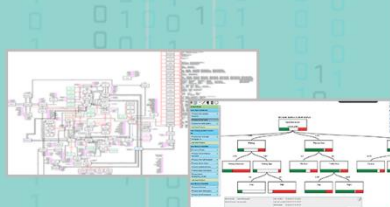
<https://www.Km4City.org>

<https://www.disit.org>

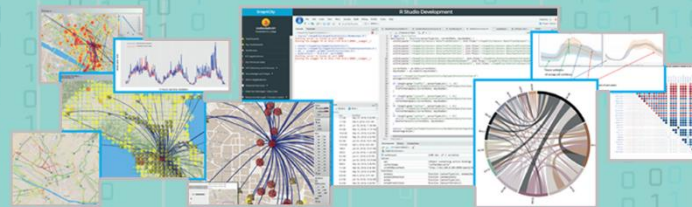




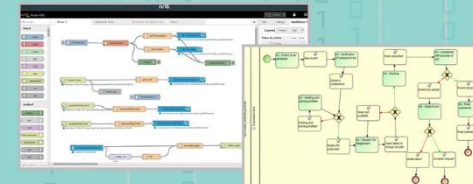
DASHBOARDS AND APPS - CONTROL ROOMS - DECISION SUPPORT SYSTEMS - WHAT-IF ANALYSIS



**EXPERT SYSTEM
KNOWLEDGE BASE
STORAGE**



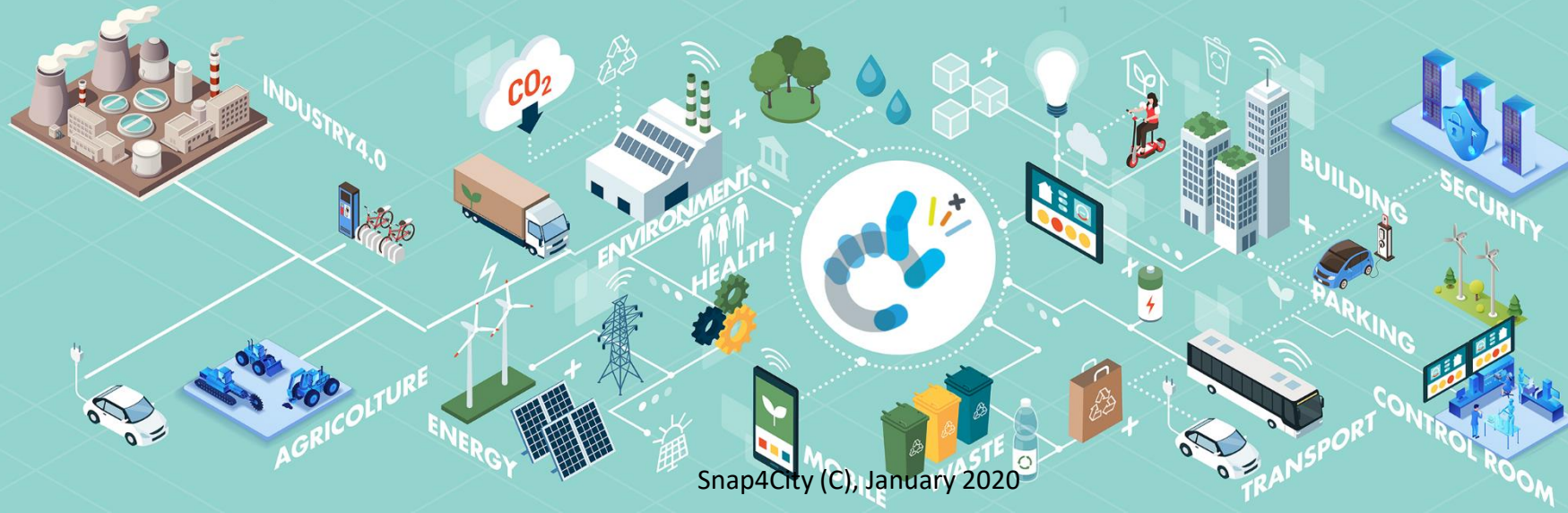
**BIG DATA ANALYTICS
ARTIFICIAL INTELLIGENCE
BUSINESS INTELLIGENCE
MACHINE LEARNING**



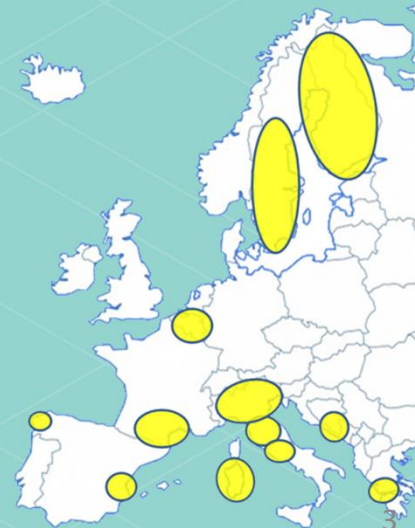
**DATA FLOWS, WORKFLOWS
MICROSERVICES
MANAGEMENT**



**METHODOLOGIES
COURSES AND COMMUNITY
LIVING LABS
DEVELOPMENT TOOLS**



Snap4City (C), January 2020

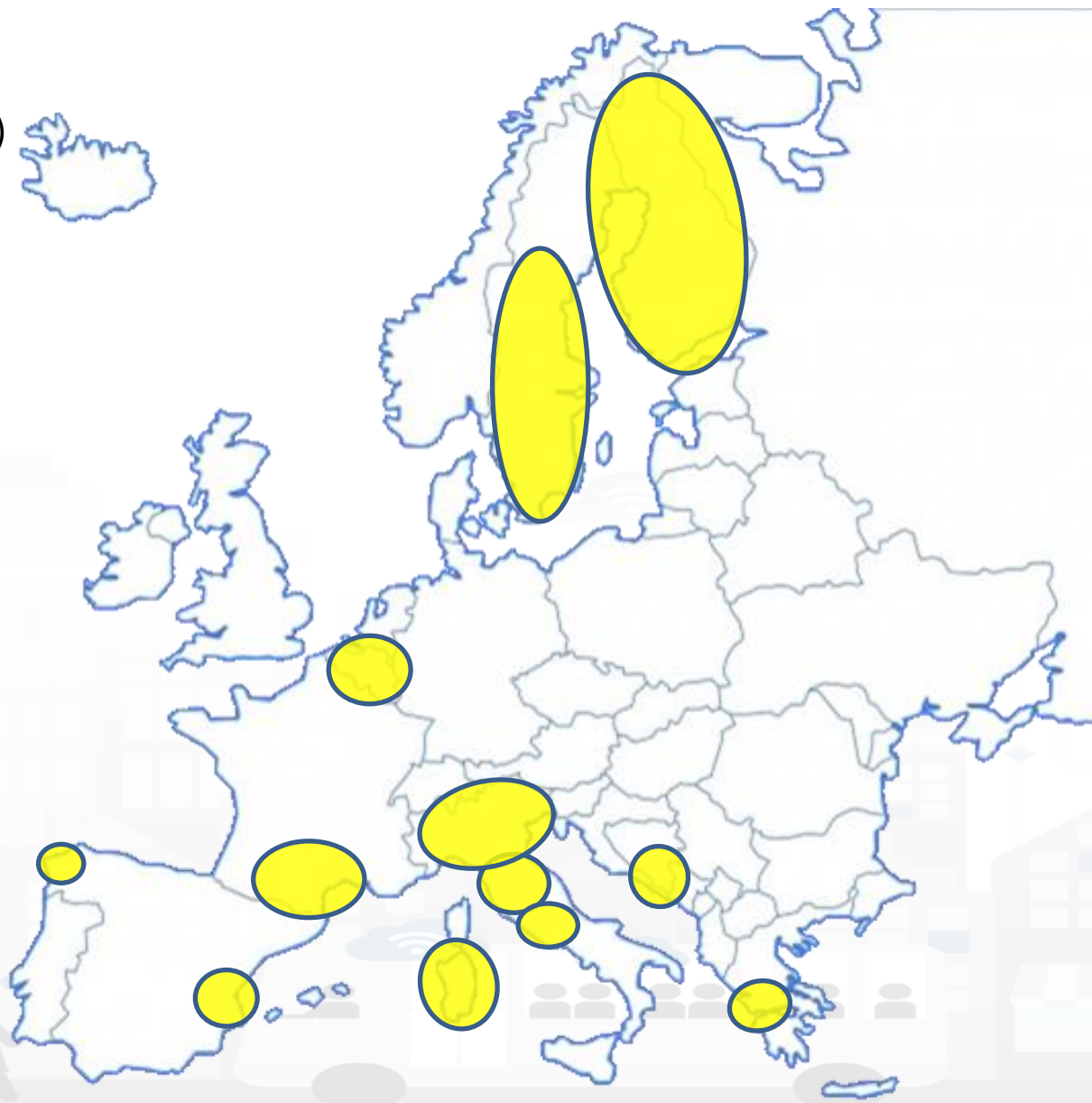


Snap4City/Industry structure

- The **Snap4xxxx** solution is released in Open Source, VM and Docker with fully support of MultiTenant/multiple-Organizations
 - 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*

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 \(Gr\)](#)

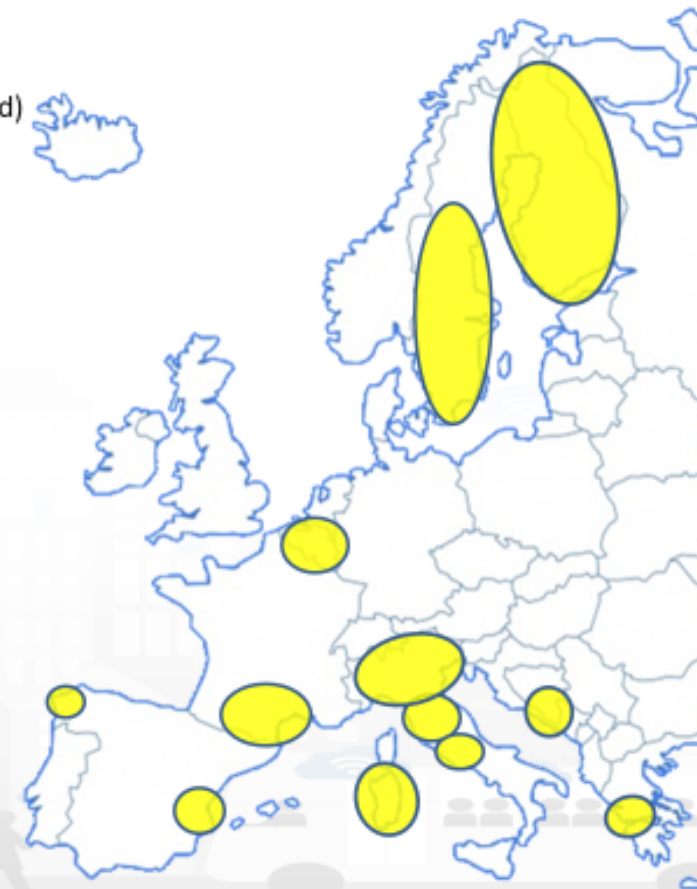


Snap4City/Industry Community

- 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.

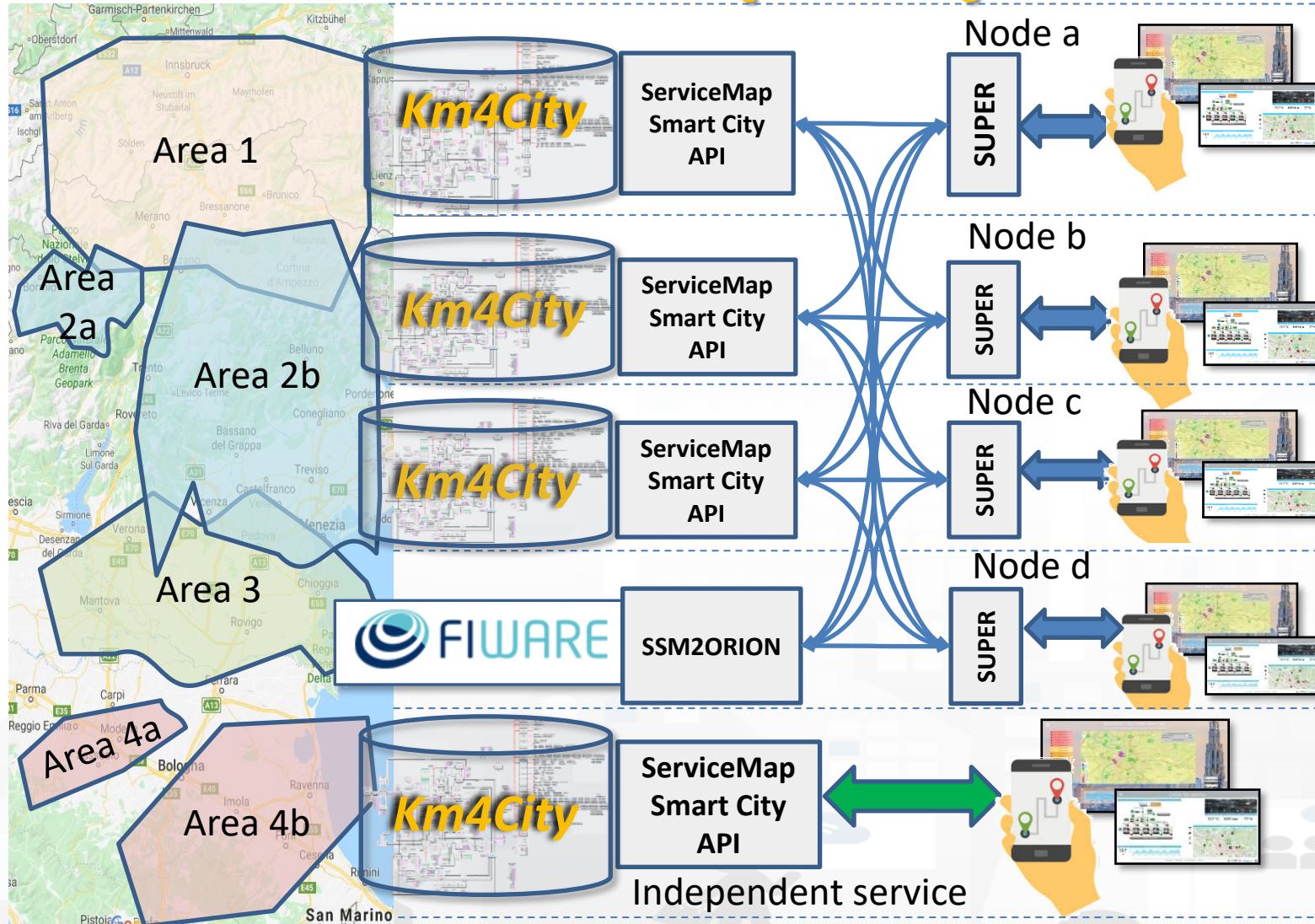
Main Organizations/areas

- [Antwerp area \(Be\)](#)
- 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 \(Gr\)](#)



Snap4City (C), October 2020

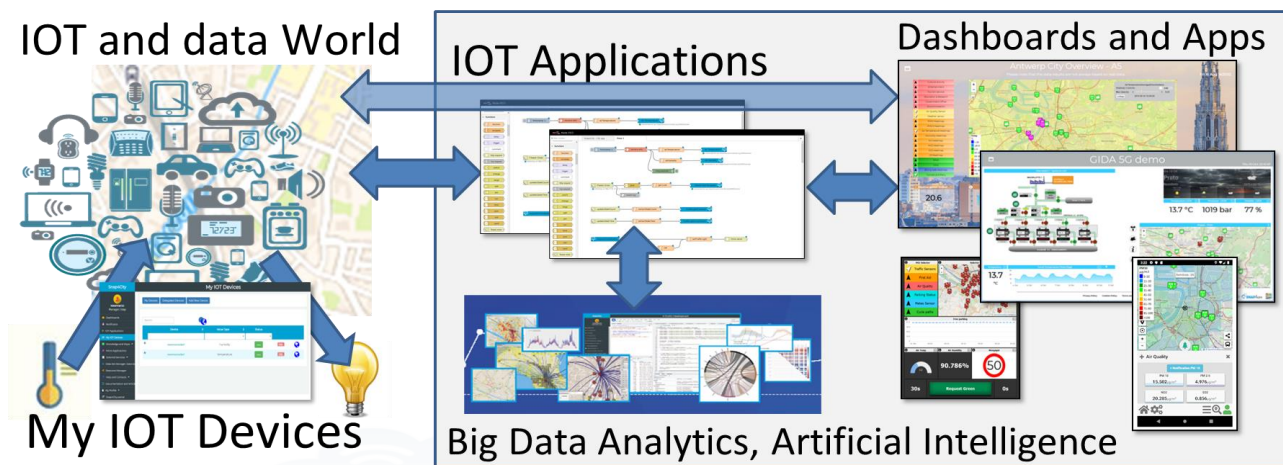
Federation of Snap4City Services



- 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 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 is not mandatory so that separate services can be produced as well
- Super and Nodes present the same Smart City APIs.

































































Free Trial

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



IF you need to go more in deep you can ask us to pass at the next Role becoming full AreaManager with full rights of development, also for Data Analytics, machine learning, etc.

On Line Training Material (free of charge)

	1st part (*)	2nd part (*)	3rd part (*)	4th part (*)	5th part (*)	6th part (*)	7th part (*)
what	General	Dashboards	IOT App, IOT Network	Data Analytics	Data Ingestion processes	System and Deploy Install	Smart City API: Web & Mob. App
PDF							
Inter active							
Video1	 	 	 	 	 	 	 
Video2	 	 	 	 	 	 	 
Video3	 	 	 	 	 	 	 
Video4	 	 	 	none	 	none	none
duration	2:55	3:16	3:41	2:00	2:48	2:35	1:47

General Overview of the full Course

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

A number of the training sections include esercitazioni

Updated versions on: <https://www.snap4city.org/577>

See also courses in ITALIANO: <https://www.snap4city.org/485>

7th Part Agenda

GO

- **Smart City API: Internal and External**

GO

- **Forging and Managing Flexible Mobile Apps, Web Apps and MicroApplications**

- Web and Mobile App with Open Development Kit
- Understanding how City User are using the City Services
- Engagement of City Users, towards a participated attitude
- Connected Drive
- Integration with Telegram: SnapBot

GO

- **Advanced Smart City API, MicroServices, Snap4City API**

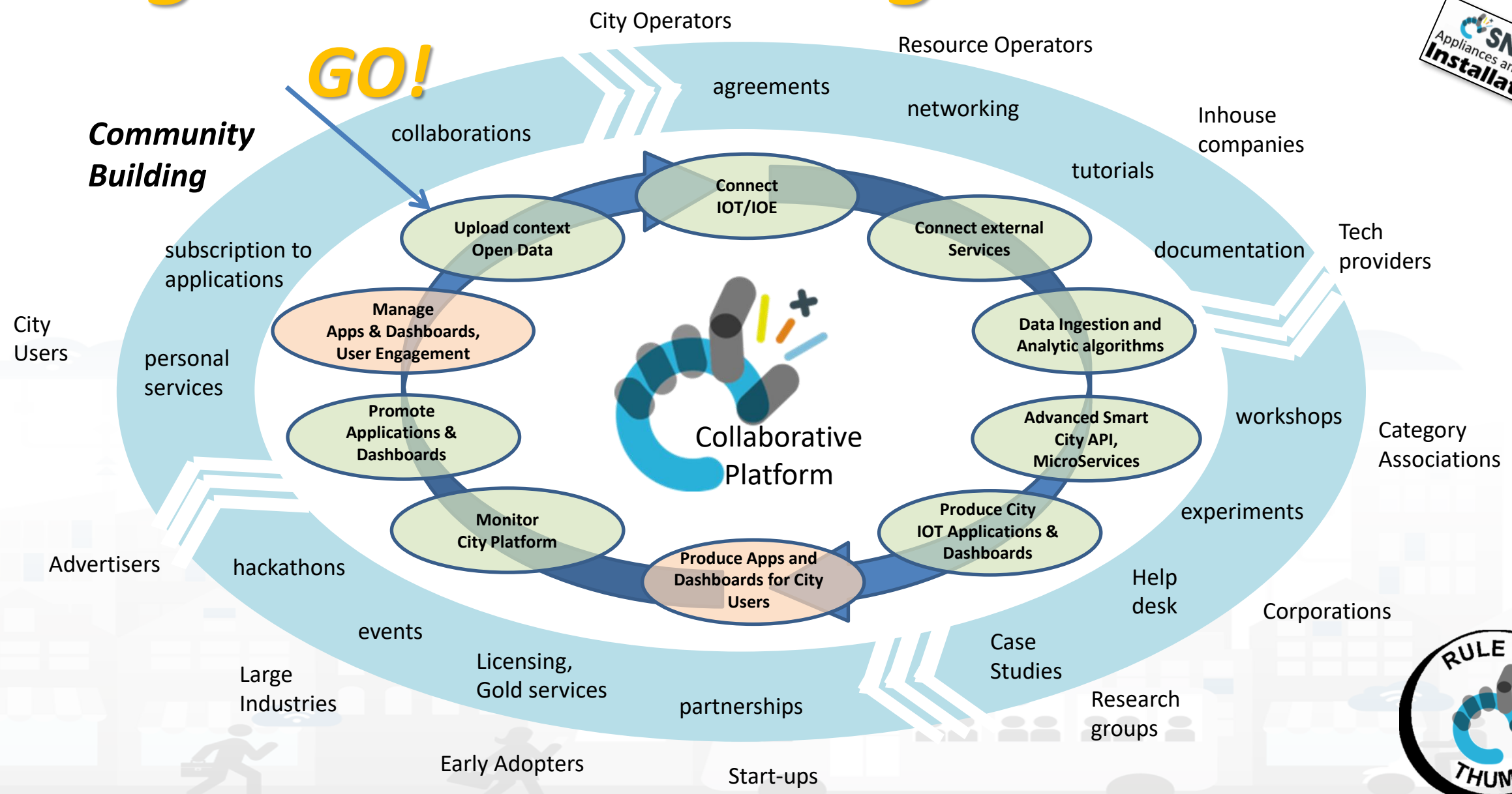
GO

- **Federated Knowledge Base and Smart City API**

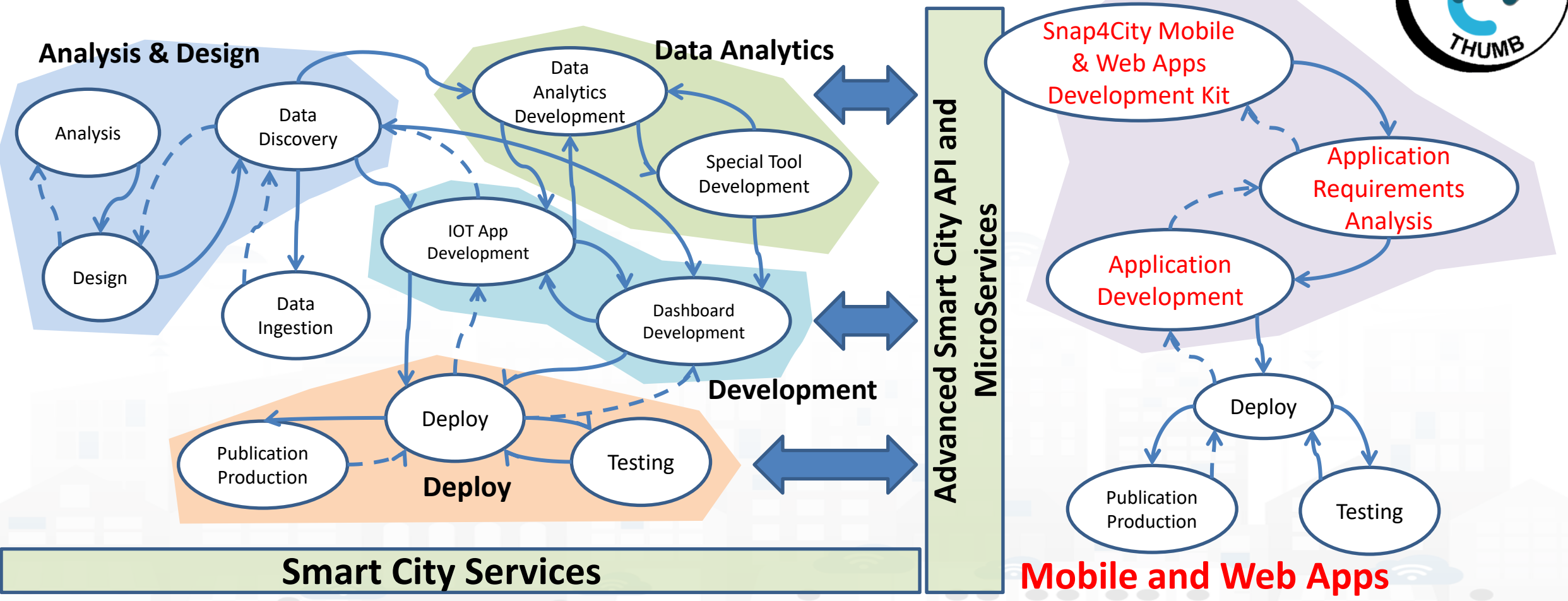
GO

- **Web and Mobile App Development Kit**
- **Acknowledgement**

Living Lab Accelerating



Develop Mobile & Web Applications Exploiting Snap4City Smart City Services



Levels of Difficulty

- Easy.
- Moderate.
- Good.
- Golden.
- Professional.
- Excellent.



non programmer level



Some JavaScript rudiment coding



JavaScript programming



Programming in R Studio



Exploiting Smart City API



Developing Full IOT Applications,
Dashboard and Mobile Apps

TOP

Smart City API: Internal and External

FROM CITY
DASHBOARD TO
APPLICATIONS

DATA GATHERING
AND CITY DATA
KNOWLEDGE
MANAGEMENT

FORGING &
MANAGING OPEN
AND FLEXIBLE WEB
AND MOBILE APPS

IOT/IOE DEVICES
AND NETWORKS

IoT APPLICATIONS
THE LOGIC AND
THE SMARTNESS

ADVANCE
SMART CITY API,
MICROSERVICES,
SNAP4CITY API

SNAP4CITY
LIVING LAB FOR
COLLABORATIVE
WORK

SNAP4CITY FOR
BEGINNERS

SNAP4CITY
ARCHITECTURE AND
ECOSYSTEM. OPENED
DEVELOPERS
AND STAKEHOLDERS

DATA ANALYTICS,
BUSINESS
INTELLIGENCE,
WHAT-IF AND
PREDICTIVE

DECISION SUPPORT
SYSTEMS AND CITY
INTELLIGENCE

TWITTER
VIGILANCE: SOCIAL
MEDIA ANALYSIS

HOW TO ADOPT
SNAP4CITY, AND
OUR ROADMAP

SNAP4CITY
AND KM4CITY
PROJECTS

SNAP4CITY THE
VIEW OF THE
ADMINISTRATORS

Internal and External Smart City API

Snap4City

User: roottooladmin1, Org: DISIT
Role: RootAdmin, Level: 7
[LOGOUT](#)

External Services

- Data Set Manager: Data Gate
- Resource Manager: Process Loader
- Development Tools
 - Web Scraping Tool
 - Web Scraping Tool (0n)
 - Web Scraping Tool (6i)
 - R Studio Development
 - R Studio Development 0.11
 - R Studio Development 0.116
 - R Studio Development TF
 - R Studio Development GFF
 - R Studio Development Gral
 - MicroServices from DataAnalytic
 - ETL Development
 - ETL Development 1
 - ETL Development 2
 - Knowledge Base Graphs
 - Knowledge Base Queries
 - Smart City API Docs: Swagger**
 - Internal API Docs: Swagger
 - Testing API by Postman
 - Source Code Access
- Management
- Settings
- User Management and Auditing
- Help and Contacts
- Documentation and Articles
- My Profile

Smart City API Docs: Swagger

swagger

Select a spec: **Advanced Smart City API**

Advanced Smart City API 1.0.0 GA53
<https://www.km4city.org/swagger/external/ascapi-openapi3.json>
SMART CITY API WEB DOCUMENTATION

Servers

Services

[GET](#) / Service discovery and information

Events

[GET](#) /events/ Event search

Locations

[GET](#) /location/ Address and geometry search by GPS

Public Transport

[GET](#) /tpl/agencies/ Agency list

[GET](#) /tpl/bus-lines/ (Bus) Lines list

[GET](#) /tpl/bus-routes/ (Bus) Routes list

Internal API Docs: Swagger

Select a spec: **IoT device registration API**

- IoT device registration API
- Notifier API
- DISCES scheduler API
- Resource Manager API
- Sensors API
- Event Logger API
- Ownership API
- Data Manager API
- Device, Broker and Value Mgmt API
- Snap4City Application API
- Engager API
- Wallet API
- User Profiler API
- My KPI API
- Snap vs Openmaint API
- Device Groups API
- Sci-Hub Processing API**

<https://www.km4city.org/swagger/external/index.html>

<https://www.km4city.org/swagger/internal/index.html>

External Smart City API

Snap4City

User: roottooladmin1, Org: DISIT
Role: RootAdmin, Level: 7
[LOGOUT](#)

External Services ▾

Data Set Manager: Data Gate

Resource Manager: Process Loader ▾

Development Tools ▴

- Web Scraping Tool
- Web Scraping Tool (0n)
- Web Scraping Tool (6l)
- R Studio Development
- R Studio Development 0.11
- R Studio Development 0.116
- R Studio Development TF
- R Studio Development GFF
- R Studio Development Gral
- MicroServices from DataAnalytic
- ETL Development
- ETL Development 1
- ETL Development 2
- Knowledge Base Graphs
- Knowledge Base Queries
- Smart City API Docs: Swagger**
- Internal API Docs: Swagger
- Testing API by Postman
- Source Code Access

Management ▾

Settings ▾

User Management and Auditing ▾

Help and Contacts ▾

Documentation and Articles ▾

My Profile ▾

Smart City API Docs: Swagger

swagger Select a spec

Advanced Smart City API 1.0.0 OAS3

<https://www.km4city.org/swagger/external/ascapi-openapi3.json>

SMART CITY API WEB DOCUMENTATION

Servers

<https://servicemap.disit.org/WebAppGrafo/api/v1>

Services ▾

[GET](#) / Service discovery and information

Events ▾

[GET](#) /events/ Event search

Locations ▾

[GET](#) /location/ Address and geometry search by GPS

Public Transport ▾

[GET](#) /tp1/agencies/ Agency list

[GET](#) /tp1/bus-lines/ (Bus) Lines list

[GET](#) /tp1/bus-routes/ (Bus) Routes list

<https://www.km4city.org/swagger/external/index.html>

External Smart City API

- **Advanced Smart City API**
 - To access the Service Map resources and query
- **Km4city Web App API**
 - To exploit MicroApplications created as tools for Dashboards, totem, web Apps, etc.
- **Orion Broker K1-K2 Authentication**
 - To communicate with IOT Orion Brokers exploiting the Secure Filter of Snap4City.
- **Heatmap**
 - To save and access to HeatMaps of the Heatmap server

Internal Snap4City API

Snap4City

User: roottooladmin1, Org: DISIT
Role: RootAdmin, Level: 7

LOGOUT

- Knowledge and Maps
- IOT Applications
- IOT Directory and Devices
- Resource Manager
- Development Tools
 - Web Scraping Tool
 - Jupyter Hub - Python
 - Web Scraping Tool (0n)
 - Web Scraping Tool (6i)
 - R Studio Development
 - R Studio Development 0.11
 - R Studio Development 0.116
 - R Studio Development TF
 - R Studio Development GFF
 - R Studio Development Gral
 - ETL Development
 - ETL Development 1
 - ETL Development 2
 - Knowledge Base Graphs
 - Knowledge Base Queries
 - Smart City API Docs: Swagger
 - Internal API Docs: Swagger**
 - Testing API by Postman
 - Source Code Access
 - How to Develop Smart Applications
- Management
 - Decision Support Systems
 - Settings
 - User Management and Auditing
 - Help and Contacts
 - Documentation and Articles
 - My Profile

Internal API Docs: Swagger

swagger Select a spec

IoT Device Registration API 2.0.0 OAS3

<https://www.km4city.org/swagger/internal/iotdeviceapi-openapi3.json>

The API accepts in input a description of an IoT device with its broker and attributes in the form of a JSON document shaped conforming to a well-defined schema operations on a graph database, also returning the URI of the inserted, updated or deleted device.

[Extended PDF documentation \(313k\)](#)

Servers

<http://www.disit.org/ServiceMap/api/v1/iot>

Registry

- POST** **/insert** For registering or updating a device in the graph database.
- POST** **/delete** For deleting a device from the graph database.
- POST** **/make-private** For marking a device as a private device.
- POST** **/make-public** For marking a device as a public device.

Device Attributes

- POST** **/disable** For disabling a device attribute.
- POST** **/enable** For enabling a device attribute that had been disabled.

Models

device >

IoT device registration API

- IoT device registration API
- Notifier API
- DISCES scheduler API
- Resource Manager API
- Sensors API
- Event Logger API
- Ownership API
- Data Manager API
- Device, Broker and Value Mgmt API
- Snap4City Application API
- Engager API
- Wallet API
- User Profiler API
- My KPI API
- Snap vs Openmaint API
- Device Groups API
- Sci-Hub Processing API**


<https://www.km4city.org/swagger/internal/index.html>

Internal Smart City API

- **IOT devices and tools API:**
 - **IoT device registration API**
 - API of the IOT Directory
 - **Sensors API**
 - API of the IOT Directory
 - **Device, Broker and Value Mgmt API**
 - API of the IOT Directory
- **Mobile App management**
 - **User Profiler API**
 - To manage the user profile for the Engager on Mobile Apps
 - **Engager API**
 - From the Engager to prepare engagements to the Mobile Apps
 - **Wallet API**
 - From the Engager to Wallet of the users of Mobile Apps and in general
 - **Snap4City Application API**

Internal Smart City API

- **Resources and entities**
 - **Snap4City Application API**
 - To manage IOT Apps
 - **My KPI API**
 - To manage MyKPI, MyPOI, POI, etc.
 - **Data Manager API**
 - **Resource Manager API**
 - To manage resources on the market place
 - **Ownership API**
 - To manage ownerships and delegations
 - **Device Groups API**
 - To manage ownerships and delegations
- **Notificator API**
- **DISCES scheduler API**
- **Event Logger API**
- **Snap vs OpenMaint API**
 - Integration with the workflow management and ticketing
- **SCI-HUB Processing API**
 - To activate data download and heatmap production from Copernicus satellite services


POSTMAN

Public Run in Postman No environment

KM4CITY API

- Introduction
- Service Discovery
- Event, POI, Address Discovery
- Service Details
- Public Transport
- User Feedbacks
- Recommender
- DISCES
- Web App
- Control Room

KM4City API

An exhaustive set of read-only APIs that have been developed in the context of the [Km4City Project](#) can be found below here.

Service Discovery

Sample calls to APIs that allow to discover the available services, and retrieve some minimal information about each of them, including the Service URI, that can be leveraged for requesting further details through calls like those that can be found in the *Details about services* section (see below).

GET Search by category in a radius

<https://servicemap.km4city.org/WebAppGrafo/api/v1/?selection=43.7909;11.2280&maxDists=0.5&categories=Accommodation&lang=en>

Search for an accommodation in a radius of 500 m from a given position, using English names & labels.

See also par. 4.2 of the [Smart City API Guidelines](#).

PARAMS	
selection	43.7909;11.2280 WGS84 latitude and longitude, that could come from a GPS device, but also could be somewhat entered by the user for gathering information about a remote location.
maxDists	0.5 Maximum distance from the given position of the services to be retrieved, expressed in kilometres. It defaults to 100 metres. If it is set to the special value "inside", services are returned whose WKT boundary contains the given position (it could be the case of a park).
categories	Accommodation The list of categories of the services to be retrieved separated by a semicolon. If omitted, all kinds of services are returned. It can contain macro categories or categories. If a macro category is specified, services are returned that belong to any of the categories in the macro category. The complete list of categories and macro categories can be retrieved on the Service Map that can be reached at https://servicemap.disit.org .

Language cURL

Example Request
 Search by category in a radius

```
curl --location --request GET "https://servicemap.km4city.org/WebAppGrafo/api/v1/?selection=43.7909;11.2280&maxDists=0.5&categories=Accommodation&lang=en"
```

Example Response
 200 - OK

```
{
  "Services": {
    "fullCount": 16,
    "type": "FeatureCollection",
    "features": [
      {
        "geometry": {
          "type": "Point",
          "coordinates": [
            11.22766494750977,
            43.7909
          ]
        }
      }
    ]
  }
}
```

<https://documenter.getpostman.com/view/4177198/km4city-api/RW83QsX5?version=latest>

Exposing Services

- **Advanced Smart City API** which can be confined into a single Smart City installation or Federated as well as for Super Service Map
 - <https://www.km4city.org/swagger/external/index.html>
- **Federated Multiple Snap4City Knowledge Bases.** This allows the creation of mobile applications that may move from multiple cities and area accessing data and making queries transparently. This solution is presently in place among the Knowledge Bases of: Antwerp/Helsinki, Tuscany/Firenze, Sardegna, etc. The resulting Service is called Super Service Map and it is integrated in the Smart City API. For example, via:
 - <https://www.disit.org/superservicemap/api/v1>
- **Federated Open Data Portals** via DataGate/CKAN that presently presents now more than 13800 data sets linked for the cities of Helsinki and Antwerp.
 - <https://datagate.snap4city.org/organization>
 - Federation, Harvesting interface is: <https://datagate.snap4city.org/harvest>
- **WFS service of Snap4City** on top of Federated Smart City API or simple Smart City API of a single ServiceMap (smart City installation). This solution permits to GIS applications and platforms (such as ArcGIS OnLine ESRI, ArcGIS Enterprise ESRI, ArcGIS Map/pro Desktop, QGIS, GeoServer, etc.) to access at Snap4City data. For Example, via:
 - <https://www.disit.org/superservicemap/api/v1/wfs>
 - <https://www.disit.org/superservicemap/api/v1/wfs?service=WFS&request=GetCapabilities&version=2.0.0>
- **WMS service of Snap4City** for publishing **maps and heatmaps**, provided by an installed GeoServer third party open source tool. For example, via:
 - <https://wmsserver.snap4city.org/geoserver/Snap4City/wms>
 - <https://www.km4city.org/swagger/external/index.html?urls.primaryName=Heatmap%20API>

TOP

FORGING & MANAGING FLEXIBLE MOBILE APPS, Web Apps and MicroApplications

FROM CITY
DASHBOARD TO
APPLICATIONS

FORGING &
MANAGING OPEN
AND FLEXIBLE WEB
AND MOBILE APPS

IoT APPLICATIONS
VS IoT EDGE
DEVICES

IoT EDGE DEVICES
IN NETWORKS

DATA COLLECTING
AND CITY DATA
KNOWLEDGE
MANAGEMENT

DATA ANALYTICS
BUSINESS
INTELLIGENCE,
WHAT-IF AND
SIMULATION

TWITTER
VIGILANCE: SOCIAL
MEDIA ANALYSIS

SNAP4CITY
LIVING LAB FOR
COLLABORATIVE
WORK

SNAP4CITY
AND KM4CITY
PROJECTS

DECISION SUPPORT
SYSTEM AND CITY
RESILIENCE

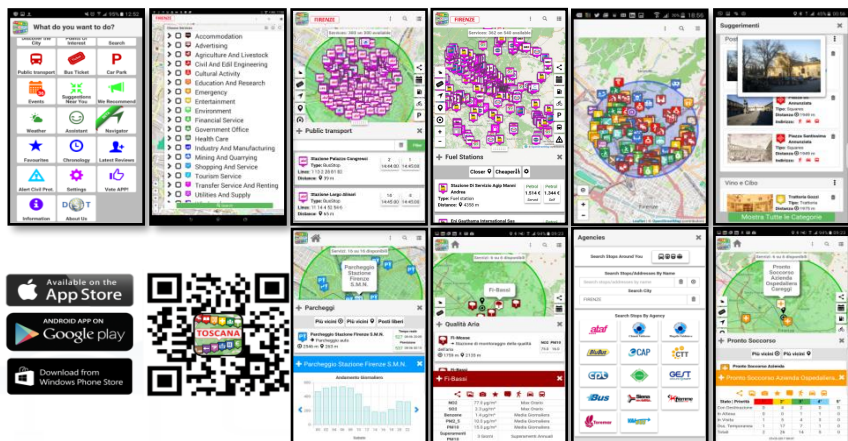
SNAP4CITY THE
VIEW OF THE
ADMINISTRATORS

SNAP4CITY
LIVING LAB FOR
COLLABORATIVE
WORK



Web and Mobile App Developers, to generate

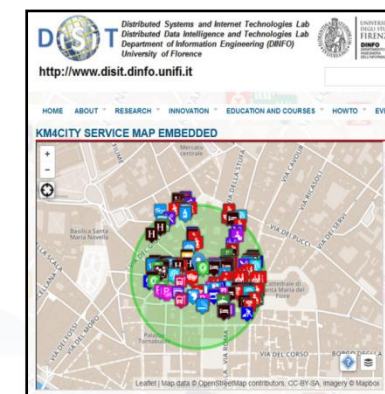
Mobile Apps



Web App HTML5



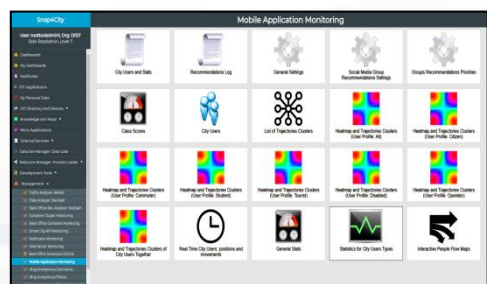
Embed into Web pages



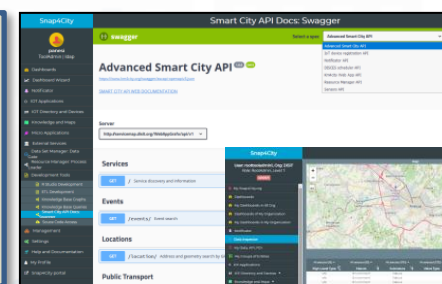
City User



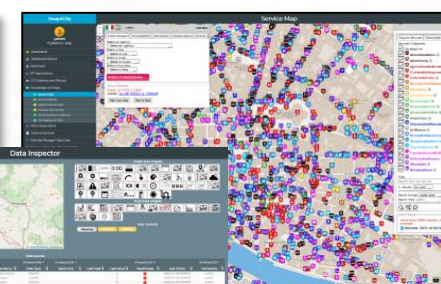
Advanced Smart City API



Km4City Open
Source
examples
dev. tool kit



Swagger



ServiceMap

Developer



Mobile Application
Monitoring
Administrator



DataInspector

Advanced SmartCity API

Swagger

- Search data: by text, near, along, etc.
 - Resolving text to GPS and formal city nodes model
- Empowering city users: contributions, suggestions, forum discussions, etc.
- Events: Entertainment, critical and mobility
- Public and Private Mobility & Transport, and predictions
- POIs, Cultural and Touristic info
- Health services and predictions
- Environmental information, heatmaps; values
- Profiled Suggestions to City Users
- Traffic flow reconstruction
- Personal Assistant: PAVAL
- User Engagement: goal experiences, and assessment
- *Sharing knowledge among cities → see Knowledge base Management*

The screenshot shows the Swagger documentation for the Advanced Smart City API. The interface includes a sidebar with navigation links and a main content area with the following sections:

- swagger**: Select a spec: Advanced Smart City API (selected), Km4City Web App API, Orion Broker K1-K2 Authentication API.
- Advanced Smart City API**: <https://www.km4city.org/swagger/externalapi-snap4city-openapi/1.json>
- SMART CITY API WEB DOCUMENTATION**
- Servers**: <https://serviceapi.digit.org/WebAppGrato/api/v1>
- Services**:
 - GET / Service discovery and information**
 - Service search near GPS position** - It allows to retrieve the set of services that are near a given GPS position. The services can be filtered as belonging to specific categories (e.g. Accommodation, Hotel, Restaurant, etc.), or having specific words in any textual field. It can also be used to find services that have a WKT spatial description that contains a specific GPS position.
 - Service search near a service** - It allows to retrieve the set of services that are near a given service identified by its serviceUri. The services can be filtered as belonging to specific categories (e.g. Accommodation, Hotel, Restaurant, etc.), or having specific words in any textual field. It can also be used to find services that have a WKT spatial description that contains a specific GPS position.
 - Service search within a GPS area** - It allows to retrieve the set of services that are inside a rectangular area. The services can be filtered as belonging to specific categories (e.g. Accommodation, Hotel, Restaurant, etc.), or having specific words in any textual field.
 - Service search within a WKT described area** - It allows to retrieve the set of services that are inside a geographic region described using the Well Known Text (WKT) format. The services can be filtered as belonging to specific categories (e.g. Accommodation, Hotel, Restaurant, etc.), or having specific words in any textual field.
 - Service search within a stored WKT described area** - It allows to retrieve the set of services that are inside a geographic region described using the Well Known Text (WKT) format, by referring to the WKT with an identifier provided when the WKT is stored. The services can be filtered as belonging to specific categories (e.g. Accommodation, Hotel, Restaurant, etc.), or having specific words in any textual field. The list of available geometries can be retrieved from the [Service Info](#) in the Search Area selection box (with Search Range specific area). New geometries can be provided using the [http://www.km4city.org/api](#) web service which can store a WKT from a shape file or providing directly the WKT string.
 - Service search by municipality** - It allows to retrieve the set of services that are in a specific municipality. The services can be filtered as belonging to specific categories (e.g. Accommodation, Hotel, Restaurant, etc.), or having specific words in any textual field.
 - Service search by query id** - It allows to retrieve the set of services associated with a query stored using the [Service Info](#) user interface.
 - Full text search** - It allows to retrieve the geolocated entities (not only services) that match with a list of keywords. The results can be possibly filtered to be within a specified distance from a GPS position, or within a rectangular area or inside a WKT geolocated area.
 - Service info** - It allows to retrieve information about a service using its serviceUri, as an HTML (format query parameter set to html) or a machine readable JSON document (format query parameter set to json).
- Parameters**:

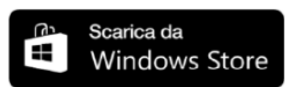
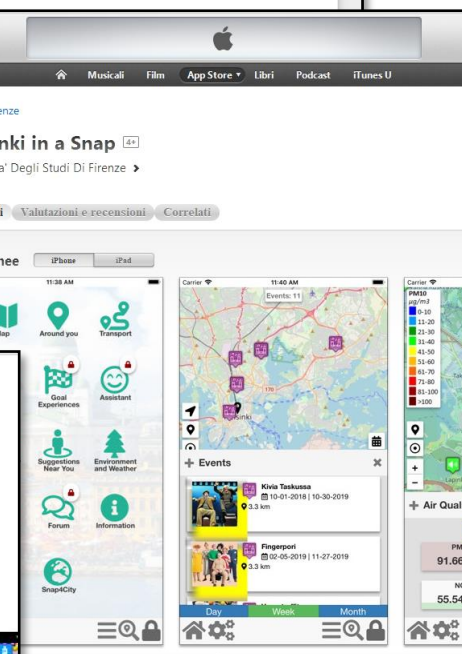
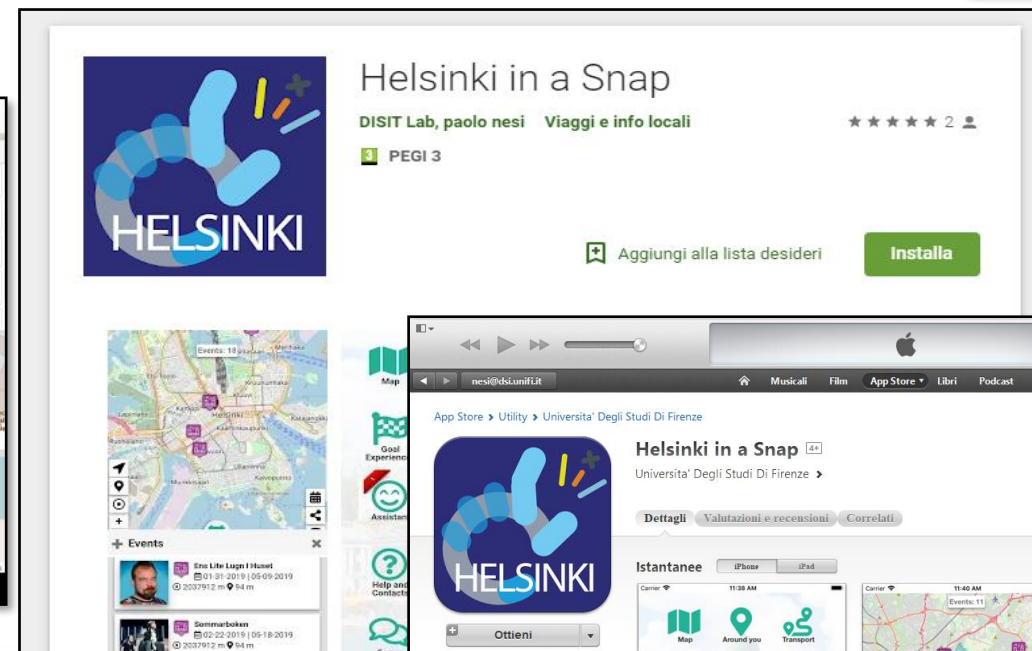
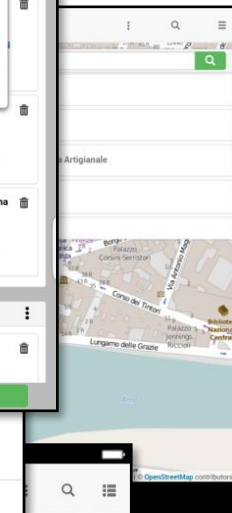
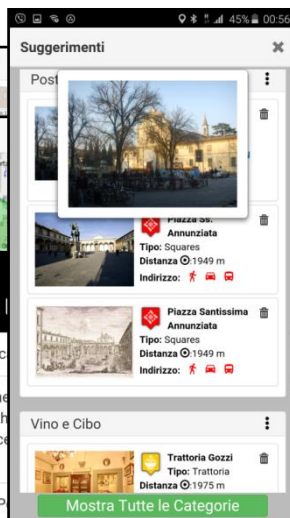
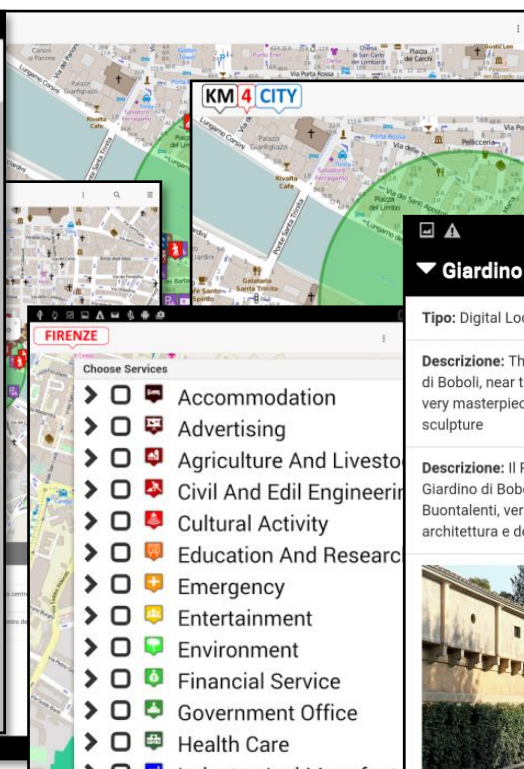
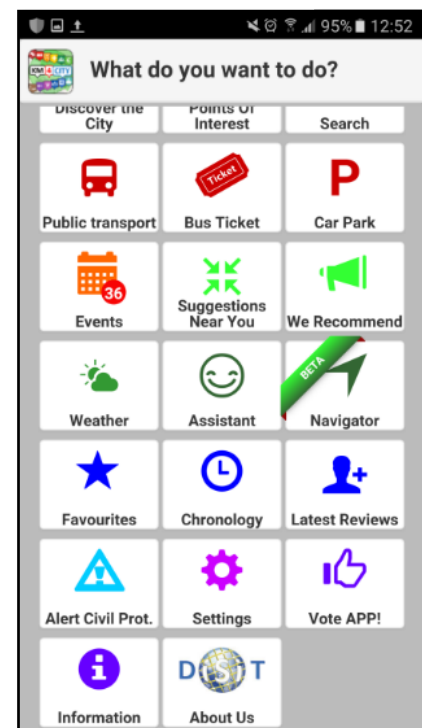
Name	Description
selection	Through this parameter, the user indicates where the services have to be searched. It could be a boundary within which to search, or a point around which to search.
string	
(query)	

TOP

Web and Mobile App with Open Development Kit



Mobile Apps



<http://www.km4city.org>

What do you want to do?

- Discover the City
- Points of Interest
- Search
- Public transport
- Bus Ticket
- Car Park
- Events
- Suggestions Near You
- We Recommend
- Weather
- Assistant
- Navigator
- Favourites
- Chronology
- Latest Reviews
- Alert Civil Prot.
- Settings
- Vote APP!
- Information
- About Us

KM4CITY

Choose Services

- Accommodation
- Advertising
- Agriculture And Livestock
- Civil And Edil Engineering
- Cultural Activity
- Education And Research
- Emergency
- Entertainment
- Environment
- Financial Service
- Government Office
- Health Care
- Industry And Manufacturing
- Mining And Quarrying
- Shopping And Service
- Tourism Service
- Transfer Service And Renting

Giardino Di Boboli

Tipo: Digital Location

Descrizione: The Prince's way ends in the Giardino di Boboli, near the Grotta del Buontalenti, that is a very masterpiece of the Mannerist architecture and sculpture

Descrizione: Il Percorso del Principe termina nel Giardino di Boboli, nei pressi della Grotta del Buontalenti, vero e proprio capolavoro dell'architettura e della scultura manierista

KM4CITY

Search: ponte

- Ponte
- Vecchio
- Vecchio

KM4CITY

Choose

- Accommodation
- Cultural Activity
- Education
- Emergency
- Entertainment
- Environment & Ag
- Financial Service
- ATM
- Bank
- Financial Institut

FIRENZE

ESERCITAZIONE MUGNONE 2016

MUGNONE 2016
28 maggio 2016

Esercitazione Mugnone 2016
28 maggio 2016
0800-1200

prevede: Firenze (FI) (ZONA: A3)

RISCHIO	TEMPI	ALLERTA
IDROGEOLOGICO IDRAULICO RETICOLO MINORE	Dalle ore 13.00 di Venerdì 27 maggio 2016 alle ore 18.00 di Venerdì 27 maggio 2016	GIALLO
IDROGEOLOGICO IDRAULICO RETICOLO MINORE	Dalle ore 18.00 di Venerdì 27 maggio 2016 alle ore 12.00 di	ARANCIONE

Suggerimenti

Post

Piazza SS. Annunziata
Tipo: Squares
Distanza: 1949 m
Indirizzo: [Icone]

Piazza Santissima Annunziata
Tipo: Squares
Distanza: 1949 m
Indirizzo: [Icone]

Trattoria Gozzi
Tipo: Trattoria
Distanza: 1975 m

Mostra Tutte le Categorie

Tutta la posta in un unico posto

Calendario

Posta

Sereno

3° 11°

0°

ROMA

Microsoft Edge

Anteprima Sk...

Tuneln Radio

Twitter

SOD&

NETFLIX

DISPONIBILE SU
Google play

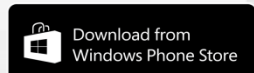
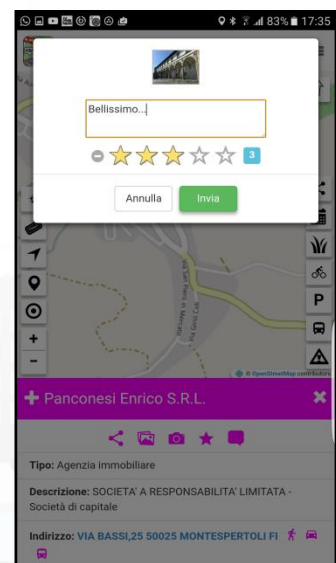
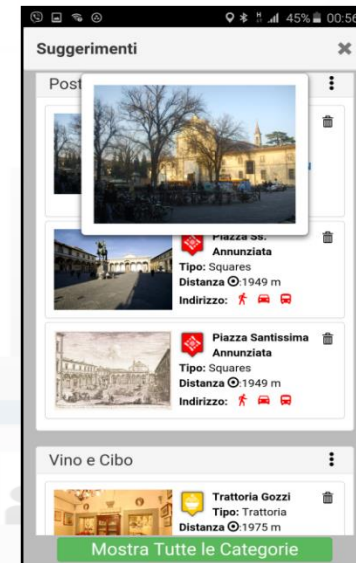
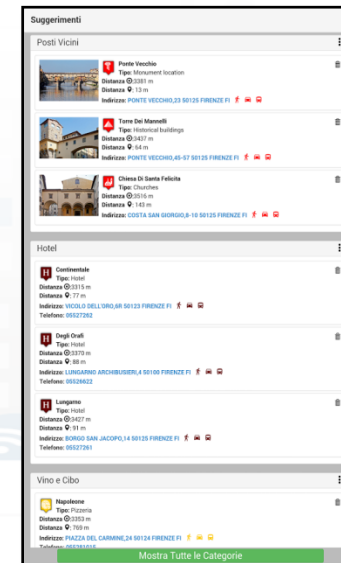
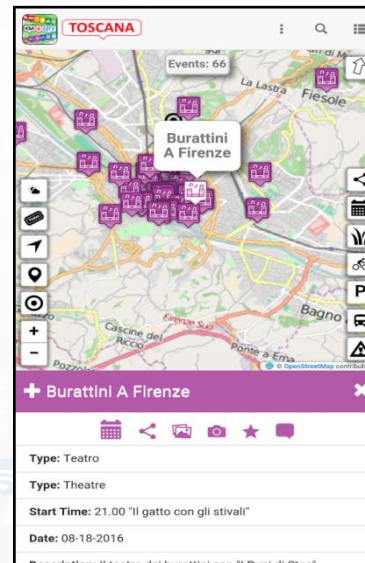
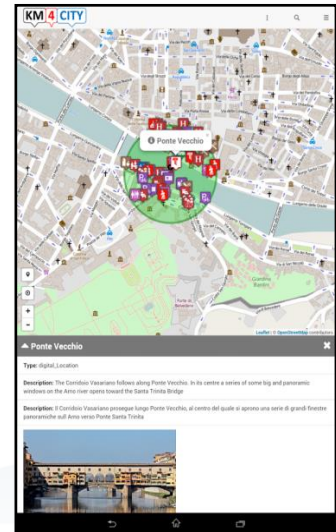
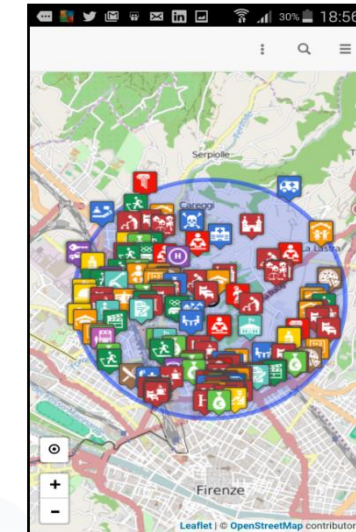
Scarica da
App Store

Scarica da
Windows Store

Km4City APP, features



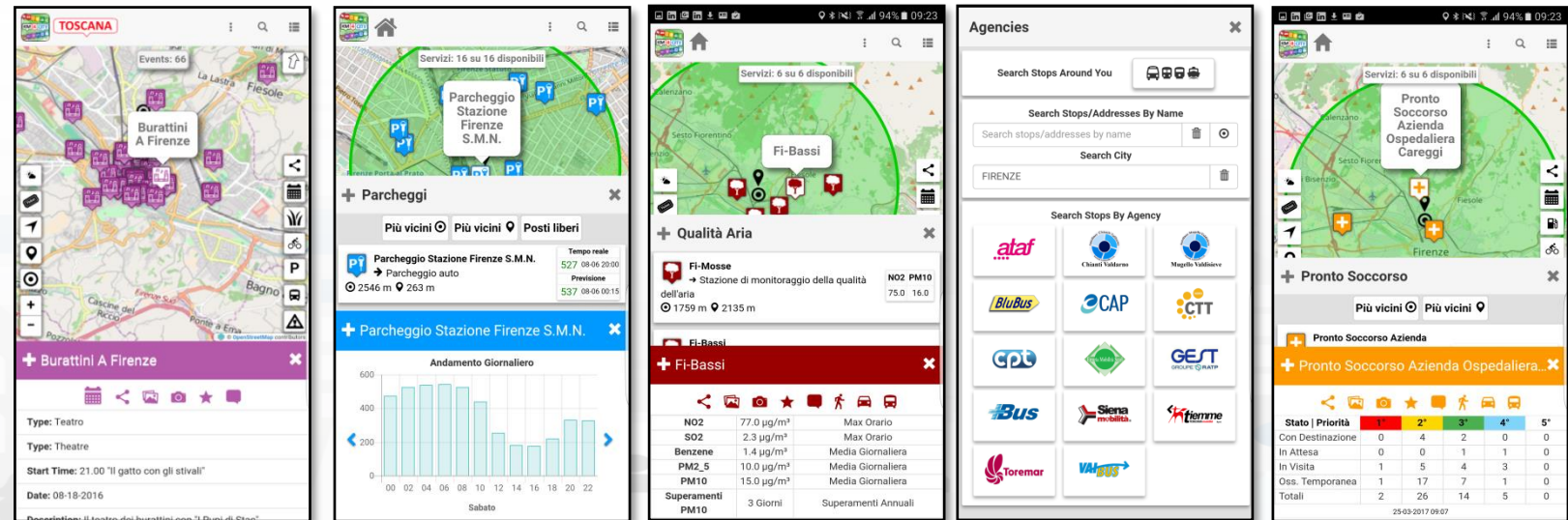
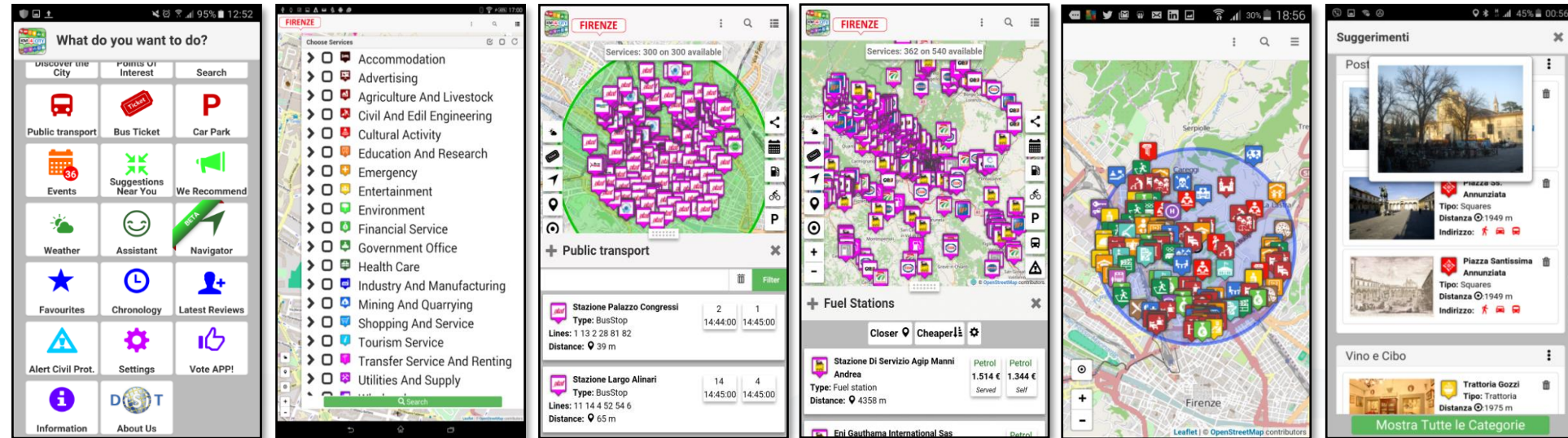
- **5 languages:** IT, EN, SP, DE, FR
- **Profiles** city users: citizens, commuter, student, tourist, operator, etc.
- **Profiled Menu** per POI
 - adaptive
- **Main Menu:** dynamic, and personalized
- **Search Text**
- **Search per POI**
 - Near to you, near to a point, a line, ...
- **Other search**
 - Close to you, events green areas, public transport, tickets, Cycling, parking, ...
 - Etc.
- **POI**
 - Preferred, Social icon
 - Ranking, Comments, Images



Km4City APP



- Smart Parking, in Tuscany
- Smart First Aid in Tuscany
- Smart Public Transportation in Tuscany
- Smart Fuel pricing in Tuscany
- Bike Sharing in Pisa
- Weather condition in Tuscany
- Environmental data
- Pollution and Pollination in Tuscany
- Traffic Sensors in Tuscany
- Smart Routing in Tuscany
- Smart Transportation in Florence
 - Events, traffic, ...
- Entertainment Events in Florence



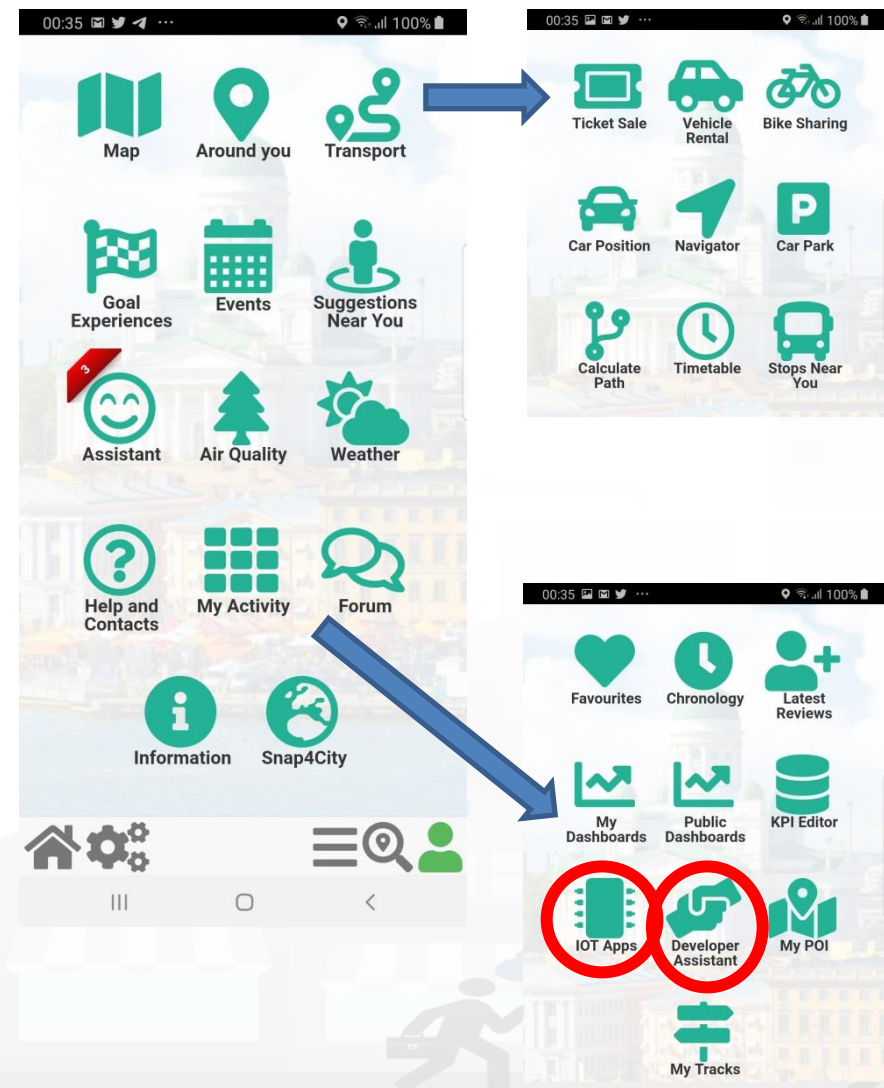
Km4City APP, features 3/3



- Navigation 3D
- Ticketing for busses
- App used as tool for city assessment
 - Wi-Fi status
 - iBeacon status
 - User behavior analysis
 - GPS movements kinds
 - OD matrix
 - International flows



Mobile App Features



- **Discovery** POI/services
- **Search:** POI, streets, suggestions
- **Mobility and transport:** Pub/priv, routing, car position, time table, park, sharing, tickets, etc.
- **Environment and Weather:** values, sensors, heatmaps, notifications
- **Assistant, Forum, Developer Assistant**
- **Goal Experiences** (Engagement)
- **Personal** data, activities, POI, tracking, IOT App, Dashboards, etc.
- **Events:** entertainment, critical
- **Sharing** position and trajectories with friends
- **Monitoring** city and personal Dashboards
- **Personalized for Operators and Developers full control of their applications on cloud**

MicroApplications

Snap4City

[LOGIN](#)

- [Dashboards \(Public\)](#)
- [Knowledge and Maps](#)
- [Micro Applications](#)
- [External Services](#)
- [Data Set Manager: Data Gate](#)
- [Resource Manager](#)
- [Development Tools](#)
- [Management](#)
- [Help and Contacts](#)
- [Documentation and Articles](#)
- [Km4City portal](#)
- [DISIT Lab portal](#)

Micro Applications

A
↓
Z
↓

Prev
1
2
3
...
Tl
Next

Q
X

[New Tab](#)

Accommodation - Antwerp in a Snap

[New Tab](#)

Accommodation - Helsinki in a Snap

[New Tab](#)

Accommodation - Toscana in a Snap

[New Tab](#)

Advertising_and_promotion - Toscana in a ...

[New Tab](#)

Agriculture And Livestock - Helsinki in a Sn...

[New Tab](#)

Agriculture And Livestock - Toscana in a Sn...

[New Tab](#)

Air Quality - Antwerp in a Snap

[New Tab](#)

Air Quality - Helsinki in a Snap

[New Tab](#)

Air Quality - Toscana in a Snap

[New Tab](#)

Air Quality Jatkasaari - Helsinki in a Snap

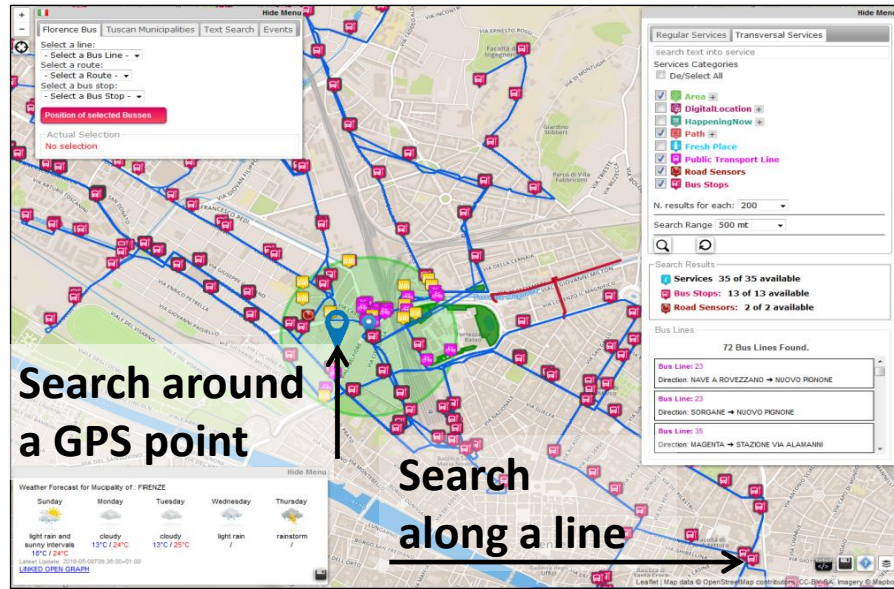
[New Tab](#)

Air Quality Sensors - Antwerp in a Snap

[New Tab](#)

Air Quality Sensors - Helsinki in a Snap

ServiceMap Dev Tool (knowledge & Map tool)

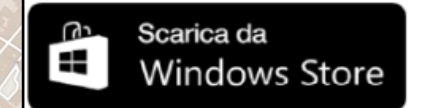


Smart City API call generation

Web App HTML5

Mobile Apps

Embed into Web pages



Advanced Smart City API

- based on Km4City engine on the back office and much more
 - Documented: <https://www.disit.org/6597>
- **ServiceMap tool** is used to visually generate/request:
 - REST Calls to exploit the Smart City APIs in web and mobile applications.
The examples of REST calls are sent by email.
 - views which can be embedded in web pages
- **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](#)

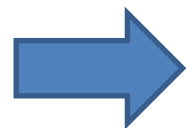
TOP

Understanding how City Users are using the City Services



The App is a Bidirectional Device

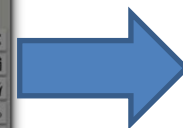
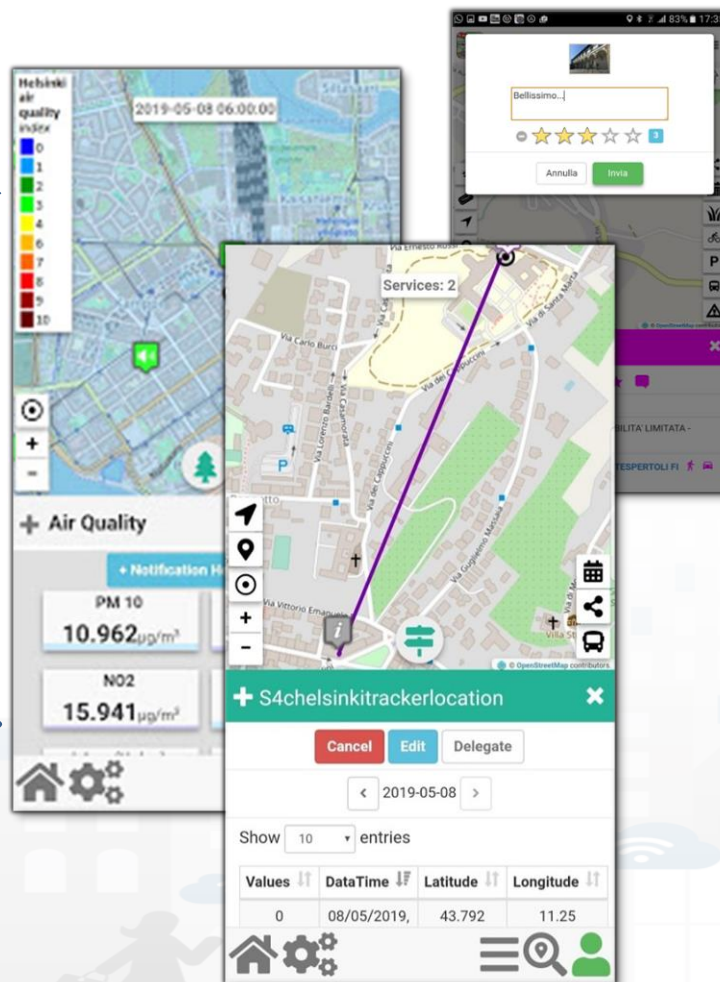
- GPS Positions
- Selections on menus
- Views of POI
- Access to Dashboards
- searched information
- Routing
- Ranks, votes
- Comments
- Images
- Subscriptions to notifications
-



Produced information

- Accepted ?
- Performed ?
- ...

Users



Derived information

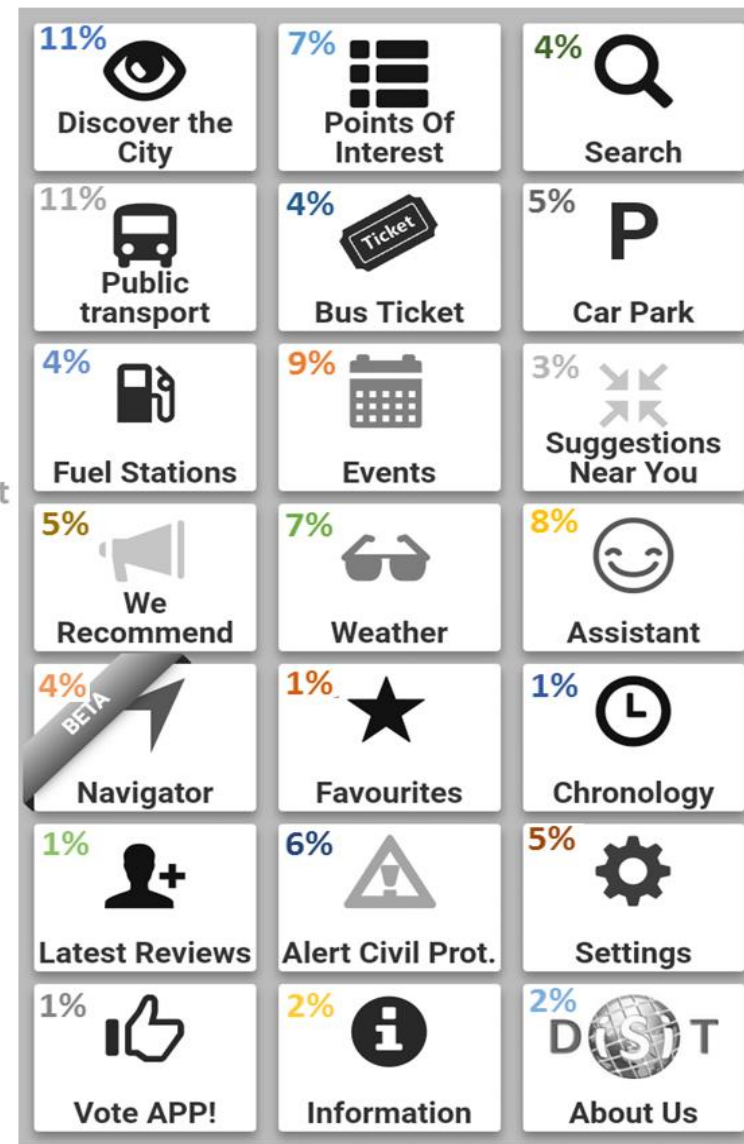
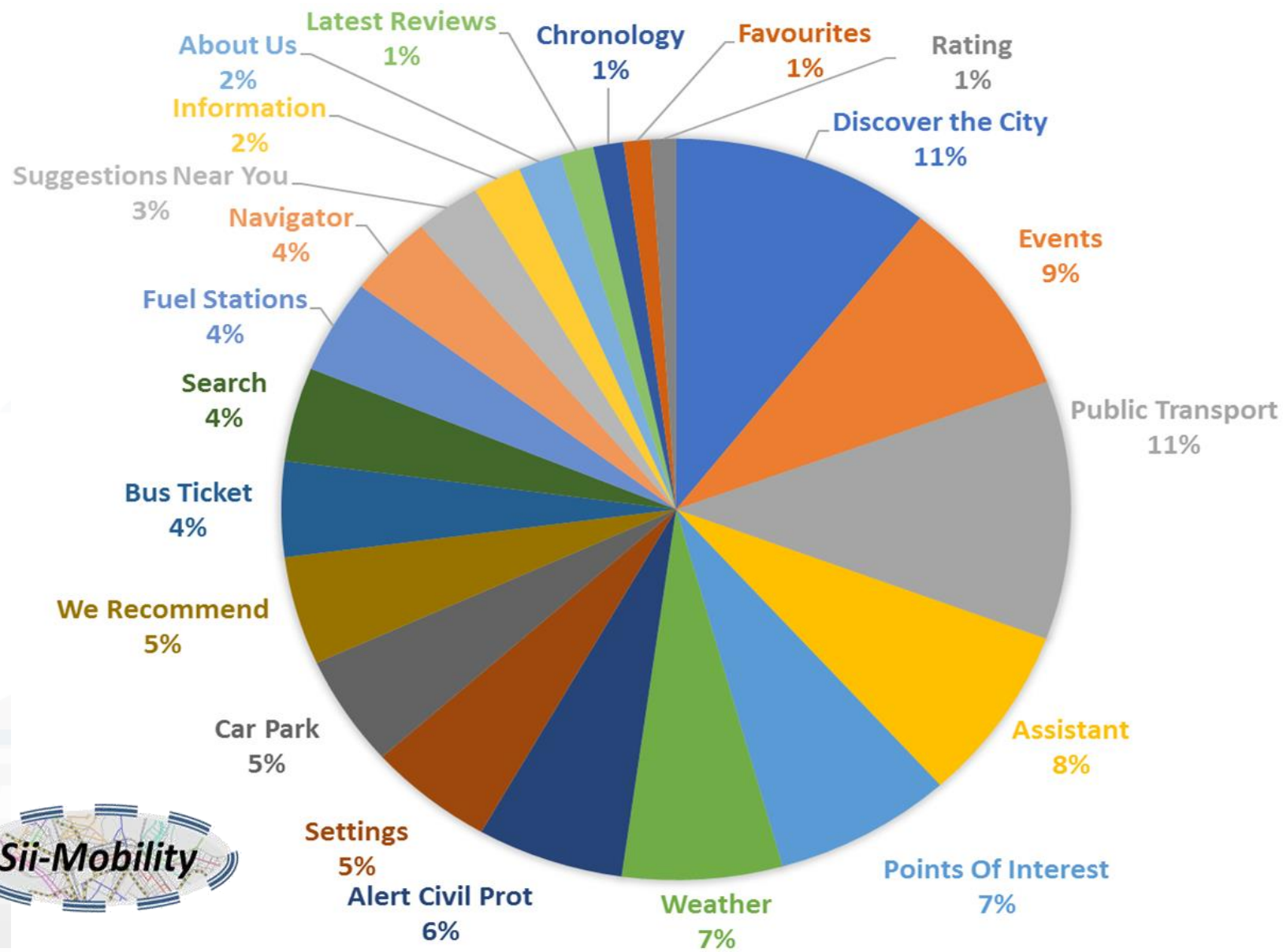
- Trajectories
- Hot Places by click and by move
- Origin destination matrices
- Most interested topics
- Most interested POI
- Delegation and relationships
- Accesses to Dashboards
- **Cumulated Scores from Actions**
- Requested information
- Routing performed
-

Produced information

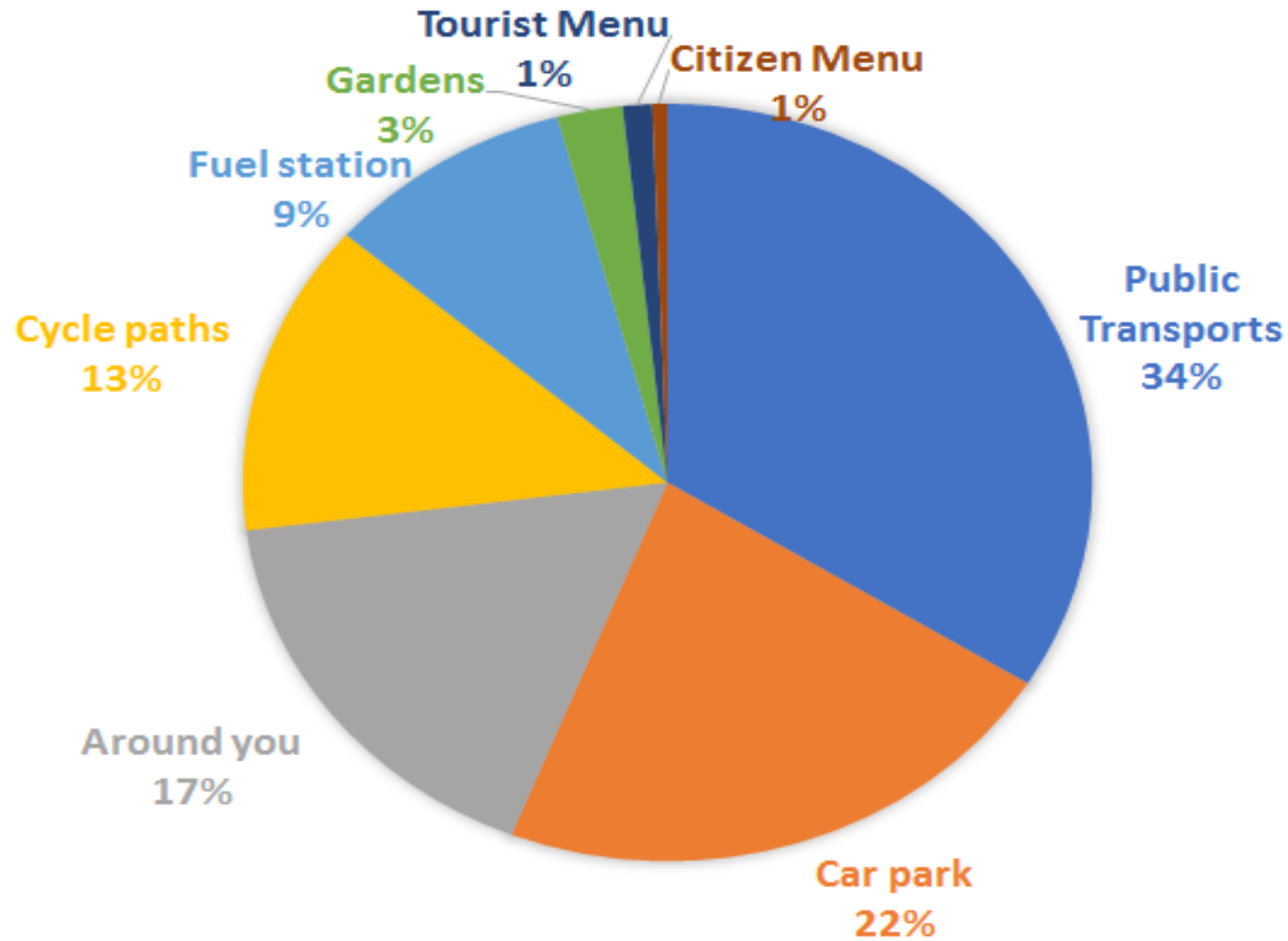
- Suggestions
- Engagements
- Notifications
- ...

System

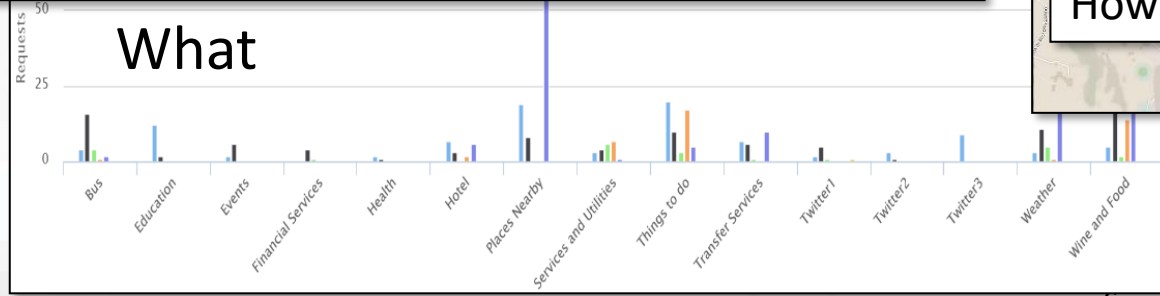
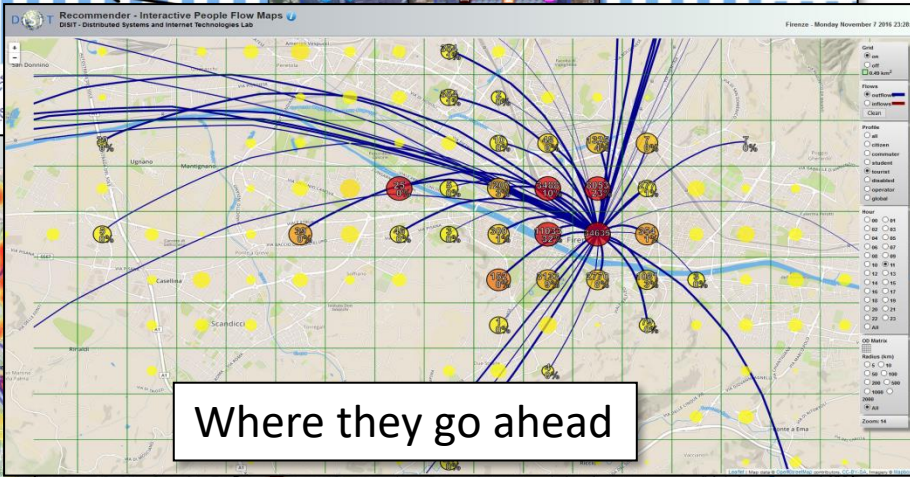
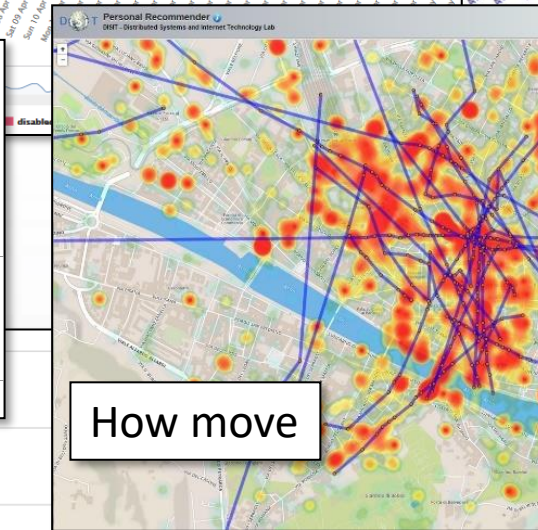
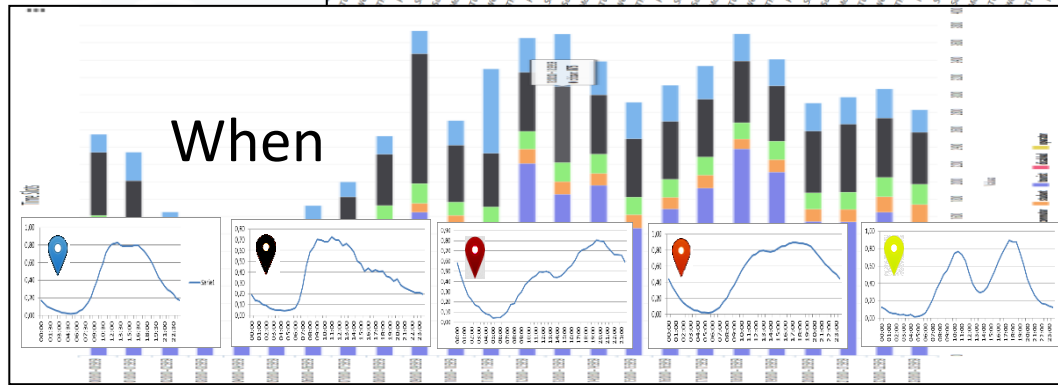
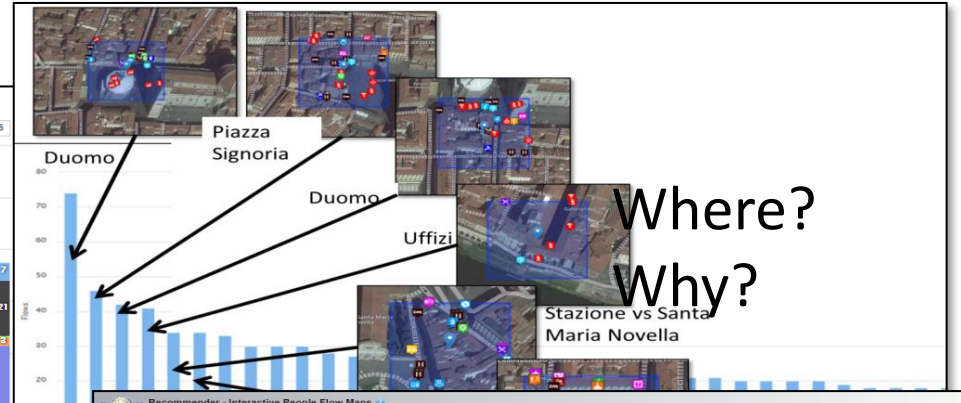
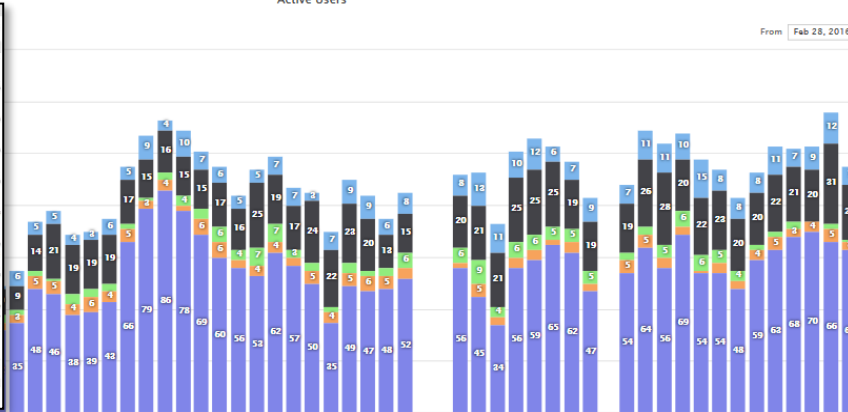
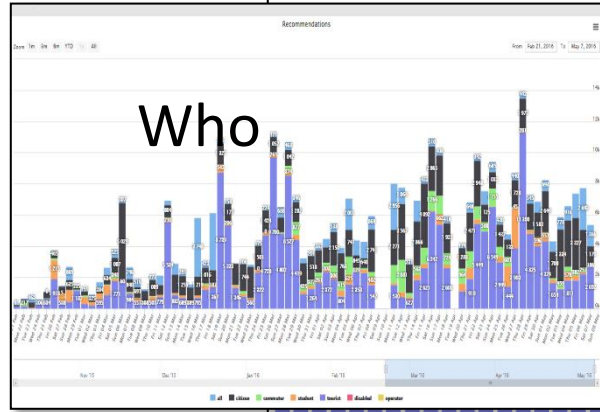
Users' preferences



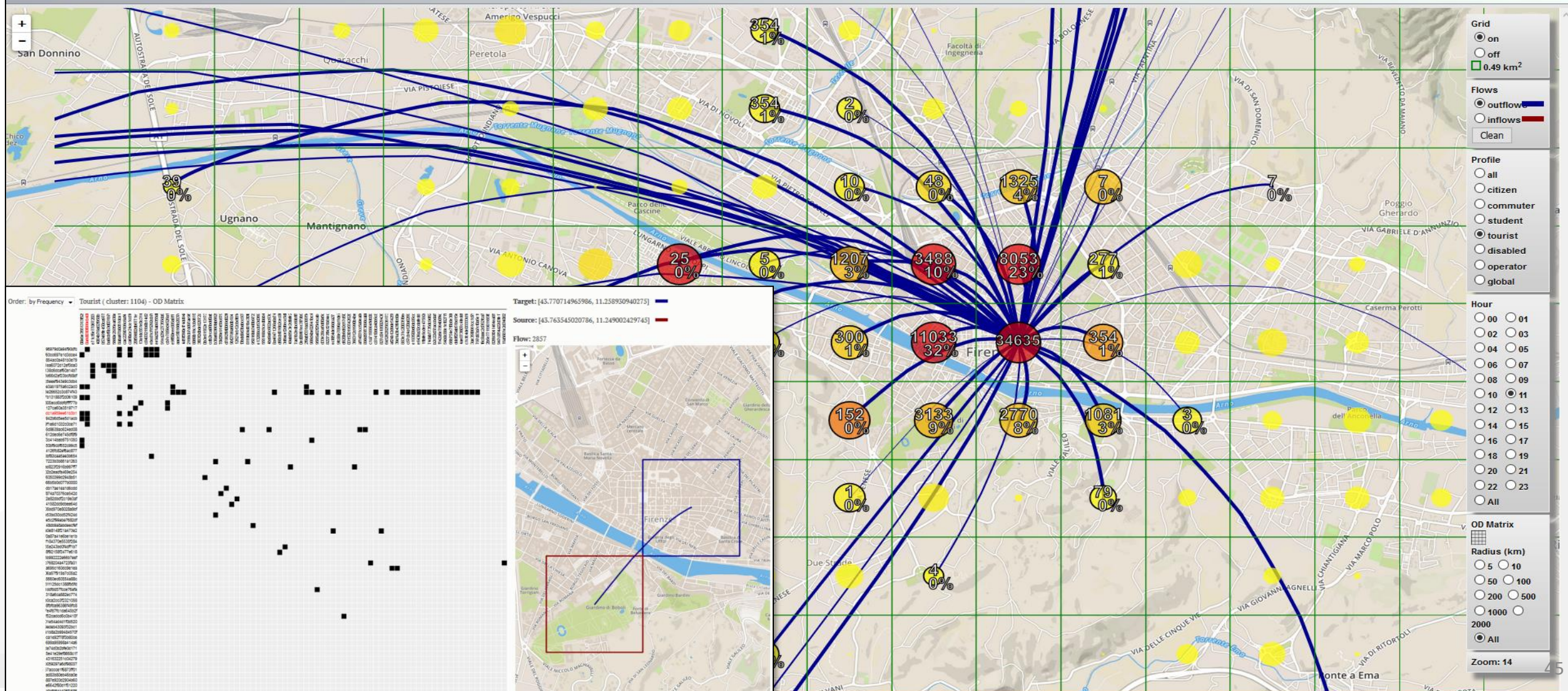
Preferred Users' Categories



User Behavior Analyser for Collective Profiling



Firenze - Monday November 7 2016 23:28:28





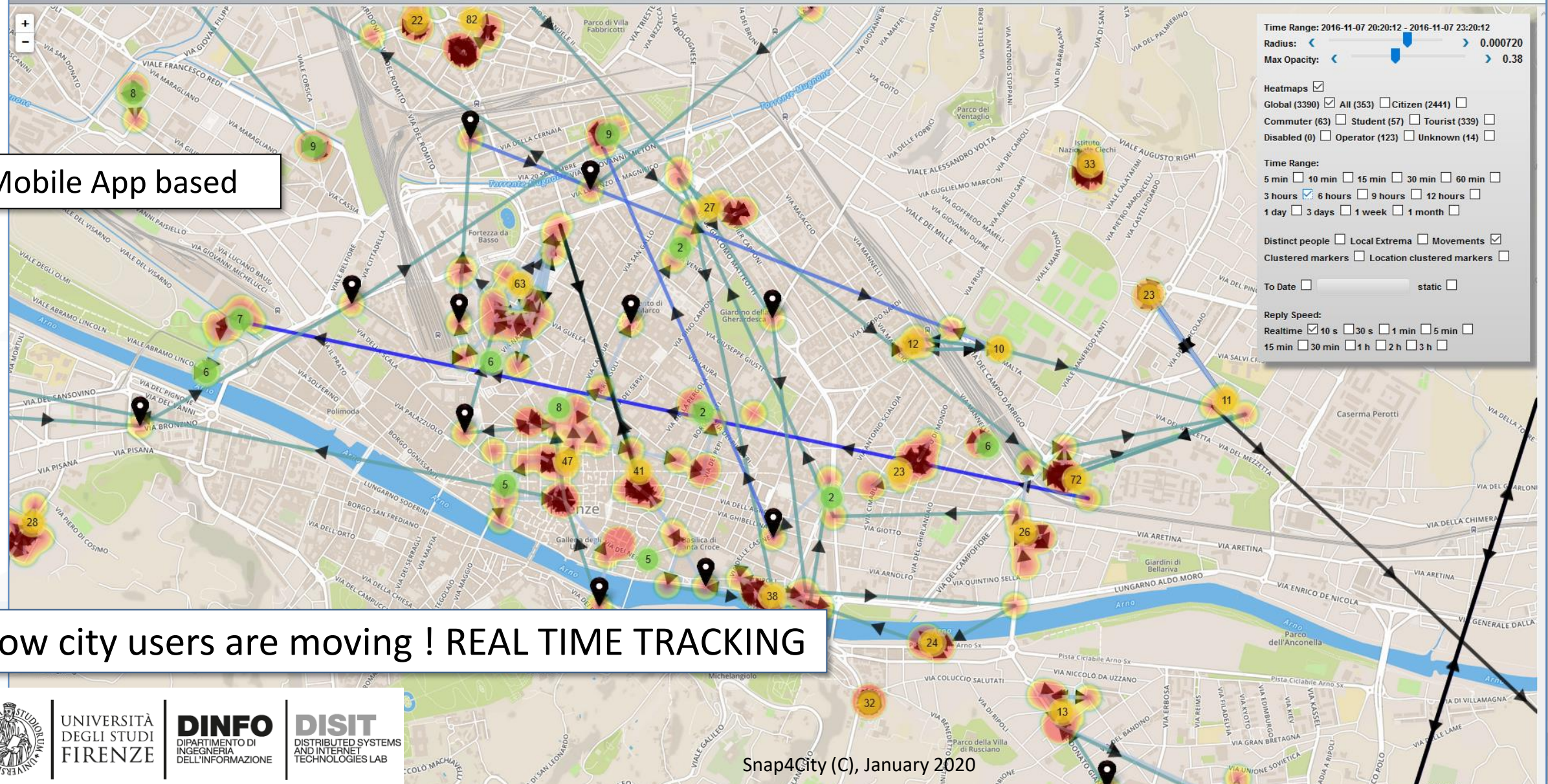
Real Time Tracking: User Behaviour Analysis



DISIT Recommender
DISIT - Distributed Systems and Internet Technologies Lab

Firenze - Monday November 7 2016 23:20:15

Mobile App based



How city users are moving ! REAL TIME TRACKING



UNIVERSITÀ
DEGLI STUDI
FIRENZE

DINFO
DIPARTIMENTO DI
INGEGNERIA
DELL'INFORMAZIONE

DISIT
DISTRIBUTED SYSTEMS
AND INTERNET
TECHNOLOGIES LAB

Snap4City (C), January 2020

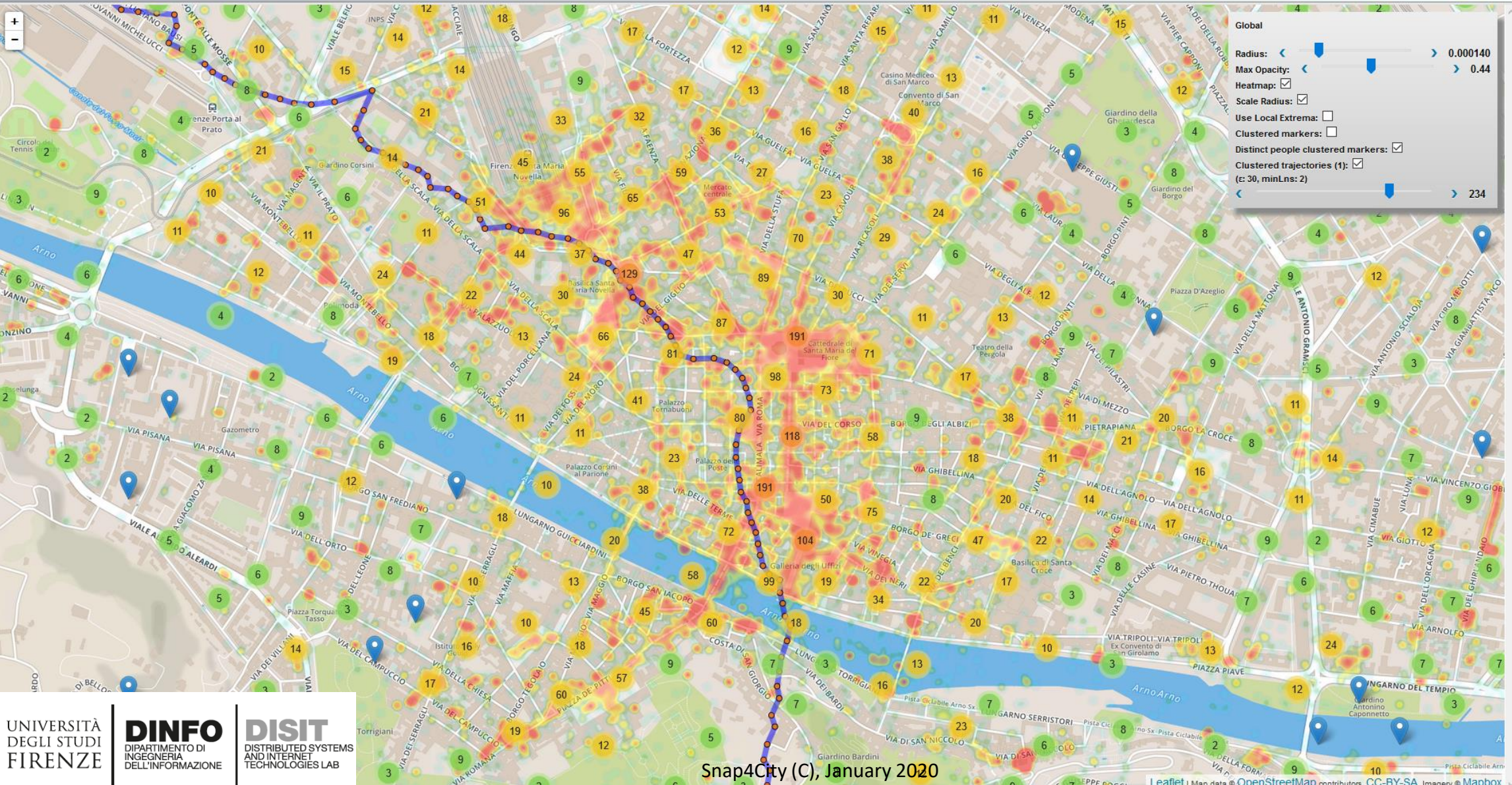
HeatMaps: Users as Sensors



DISIT Recommender - Heatmap and Trajectories Clusters of City Users Together

DISIT - Distributed Systems and Internet Technologies Lab

Firenze - Sunday January 8 2017 13:57:00



Snap4City (C), January 2020

Leaflet | Map data © OpenStreetMap contributors, CC-BY-SA, Imagery © Mapbox



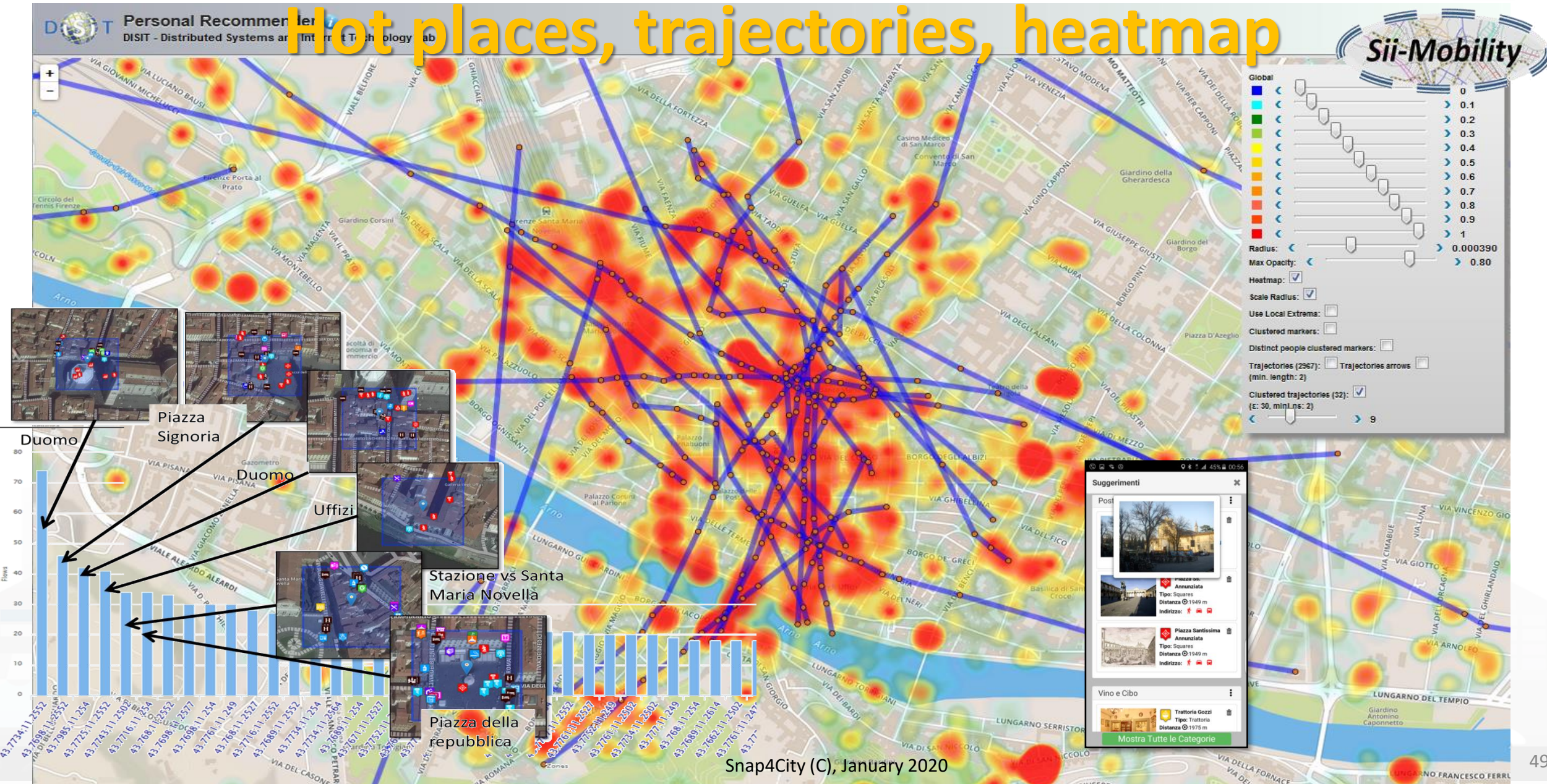
UNIVERSITÀ
DEGLI STUDI
FIRENZE

DINFO
DIPARTIMENTO DI
INGEGNERIA
DELL'INFORMAZIONE

DISIT
DISTRIBUTED SYSTEMS
AND INTERNET
TECHNOLOGIES LAB

User Behaviour Analyser

Hot places, trajectories, heatmap



Understanding City User Behaviour

- **Mobile Applications** can send data via Advanced Smart City API to collect data about the city usage by the city users via a signed consent
 - See Mobile and Web App: Toscana in a Snap, Helsinki in a Snap, Antwerp in a Snap.
- **City User behavior analysis** includes production of:
 - suggestions, trajectories, hot places/heatmaps, etc.
 - origin destination matrices
 - data for the city user engagement
 - Etc.



<https://www.snap4city.org/drupal/node/489>

TOP

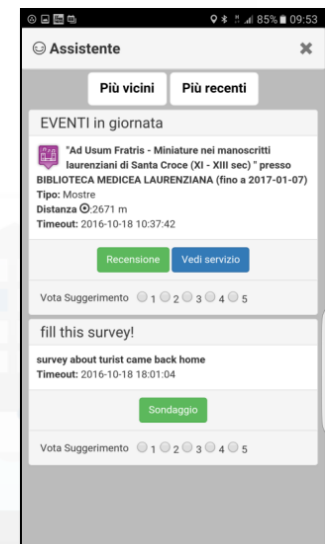
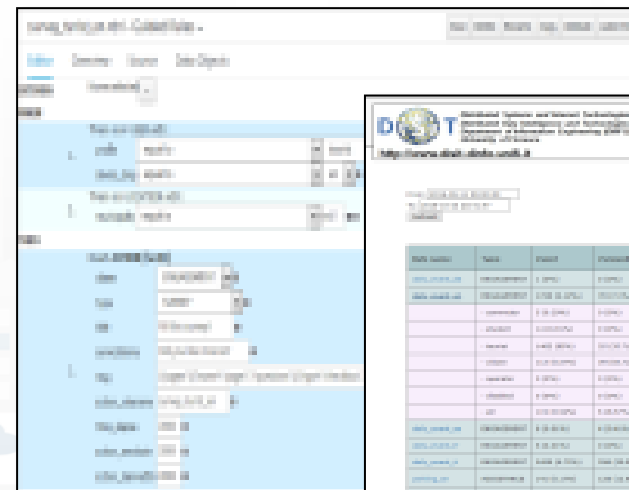
Engaging City Users Towards Virtuous Participated Attitude



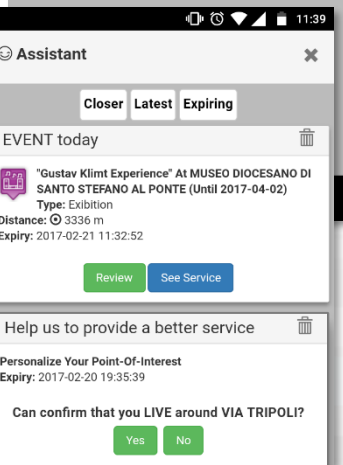
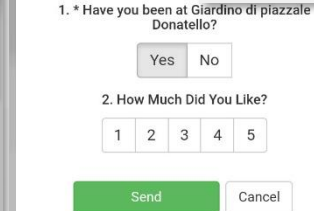
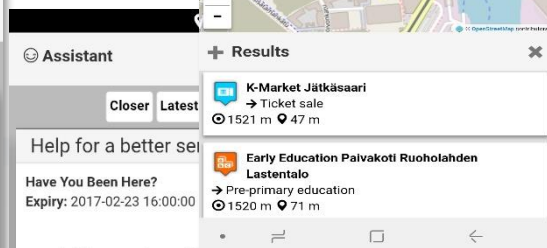
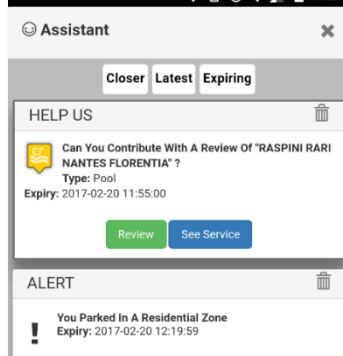
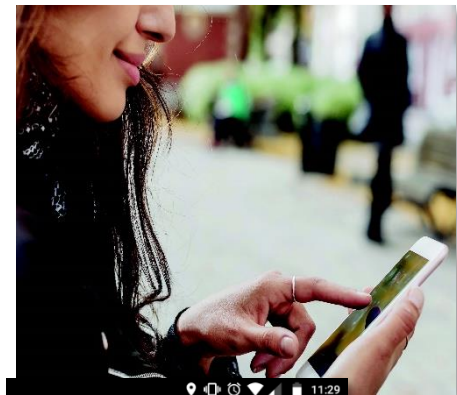


Profiled Engagements to City Users

- 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

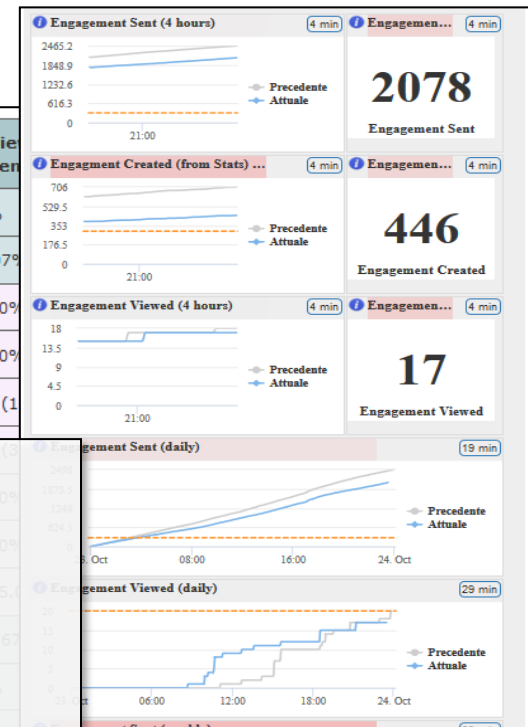


User
context

City
context

Rules

Rule name	Type	#sent	#viewed	#viewed / #sent
daily_event_de	ENGAGEMENT	1 (0%)	0 (0%)	0%
daily_event_en	ENGAGEMENT	1720 (2.12%)	70 (7.1%)	4.07%
- commuter		5 (0.29%)	0 (0%)	0 (0%)
- student		14 (0.81%)	0 (0%)	0 (0%)
- tourist		1462 (85%)	25 (35.71%)	25 (17.1%)



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 by a bus ticket

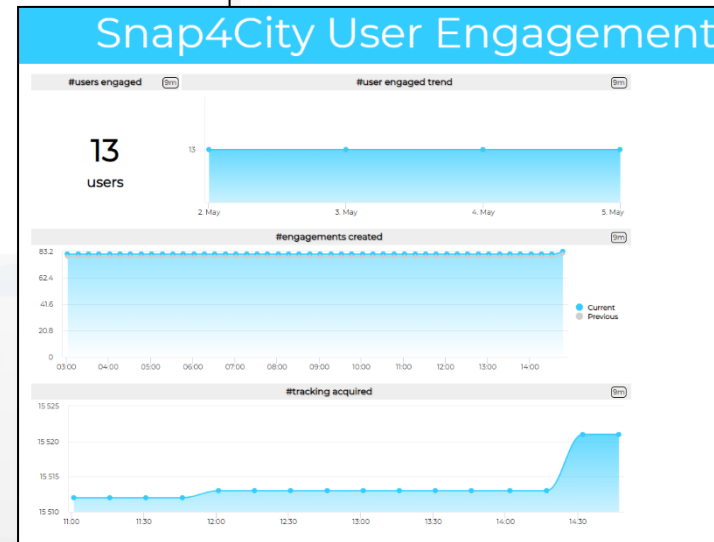
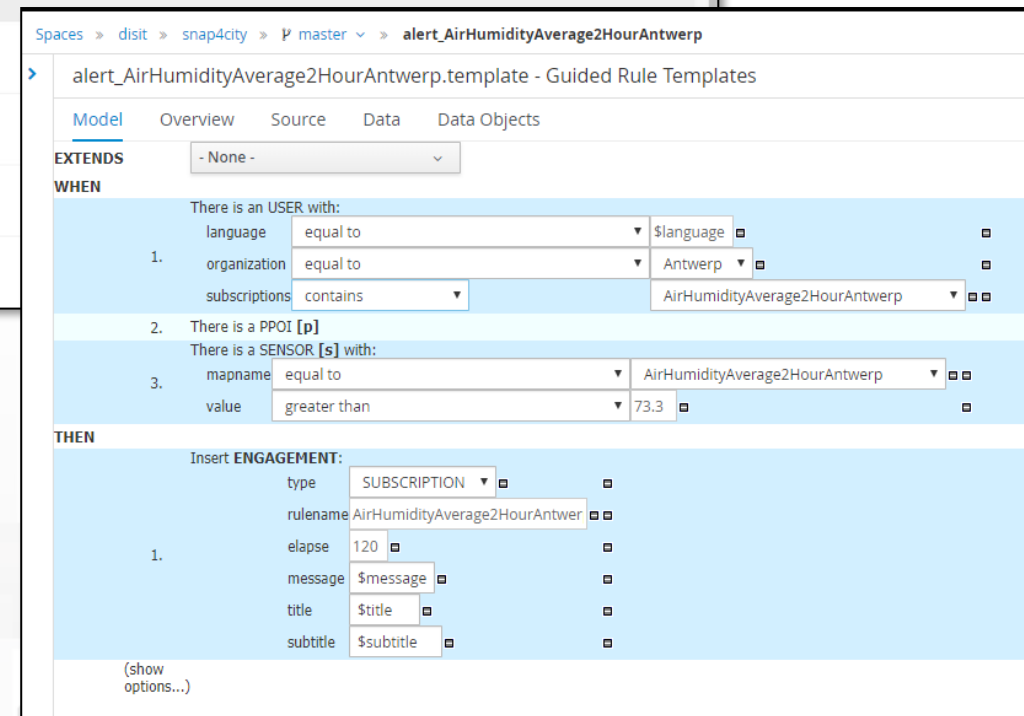
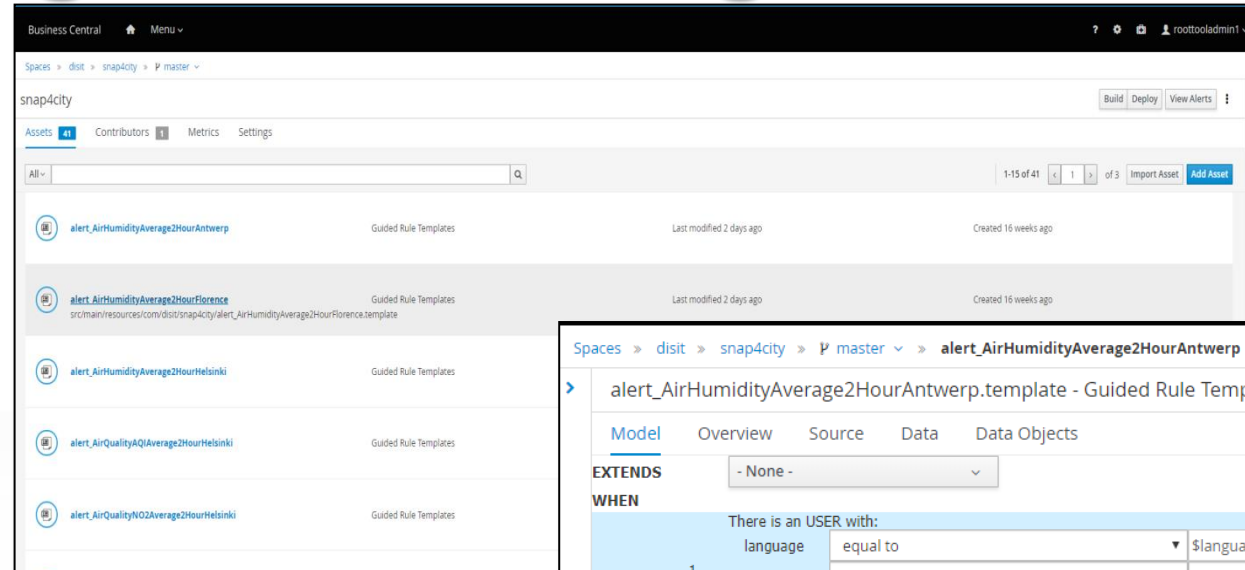
Suggest (in Italian) an event currently on in the city
Alert (in English) if the user parked in a residential zone
Alert (in Spanish) if the user parked in a residential zone
Alert (in Italian) if the user parked in a residential zone
Ask (in German) a contribution for a nearby service

Engaging City Users

- **Mobile Applications** can use Advanced Smart City API to collect data about the city usage by the city users via a signed consent
- It can be used for sending engagements to them such as to:
 - **Inform**
 - You have parked out of your residential parking zone
 - The Road cleaning is this night
 - The waste in S.Andreas Road is full
 - **Engage**
 - Please Provide a comment, a score, etc.
 - **Stimulate / recommend**
 - Events in the city, services you may be interested, etc..
 - **Provide Bonus**
 - Since you have parked here you can get 1 Bonus
 - We suggest you to leave the car out of the city, this bonus can be used to buy a bus ticket

Engagement Manager

- Definition of Rules for campaigns
- Monitoring and follow-up for each City
- Segmented for user kind and interest



Sii smart. Sii-Mobility!

Scarica

Dal 15 aprile al 15 luglio scegliere il trasporto pubblico ti premia! Scarica l'app "Toscana dove, cosa", guadagna punti viaggiando in autobus e vinci tanti fantastici premi! Per maggiori informazioni visita il sito info.sii-mobility.org

In palio per te
Carnet multicorsa Cap e
voucher per:

Sii smart. Sii-Mobility! Scarica, viaggia, vinci!

Dal 15 aprile al 15 luglio scegliere il trasporto pubblico ti premia! Scarica l'app "Toscana dove, cosa", guadagna punti viaggiando in autobus e vinci tanti fantastici premi! Per maggiori informazioni visita il sito info.sii-mobility.org



UNIVERSITÀ
DEGLI STUDI
FIRENZE

DINFO
DIPARTIMENTO DI
INFORMAZIONE

DISIT
DISTRIBUTED SYSTEMS
AND INFORMATION
TECHNOLOGIES LAB

In palio per te
Carnet multicorsa Cpt e
voucher per:



Ci Prendiamo cura del tuo benessere



Sii smart. Sii-Mobility! Scarica, viaggia, vinci!



Dal 15 aprile al 15 luglio scegliere il trasporto pubblico ti premia! Scarica l'app "Toscana dove, cosa", guadagna punti viaggiando in autobus e vinci tanti fantastici premi. Per maggiori informazioni visita il sito info.sii-mobility.org



UNIVERSITÀ
DEGLI STUDI
FIRENZE

DINFO
DIPARTIMENTO DI
INFORMAZIONE

DISIT
DISTRIBUTED SYSTEMS
AND INFORMATION
TECHNOLOGIES LAB

Campaing on Sustainable Mobility

Rules for Rewards

- **ASSISTANCE**

- If public transport is detected after bus line suggestion on trajectory usually made on private transport → 10points
 - Why don't you take the bus line 4 in Piazza Marconi to reach your workplace? You save money, you respect the environment and you will be stress free for not worry about parking!
- Once a day, if public transport is detected after suggestion on an alternative bus line availability → 3points
 - Why don't you take the bus line 4 that stop just 50 meters far from you? You save money, you respect the environment and you will be stress free for the traffic jam!
- If public transport is detected for at least 30(?) minutes a day → 1point

- **ENGAGEMENT**

- Survey on commuter and their preferred way of mobility → 1point
 - How many minutes you usually commute to go to work?
How do you rate the service?
- Feedback on public transport → 1point
 - Which current public transport are you using? Are the service in line with your expectation?
- Comments/Photo/Rate or survey on POI (public transport) → 1point
- Survey on use of the App after N days or for tourist coming home → 1point
- Feedback on PPOI or mobility → 1point

Current Numbers

From 1° September 2016

– Detected 2108 PPOIs on 1080 users

- 437 HOME
- 285 WORK
- 34 SCHOOL
- 1350 EXTRA

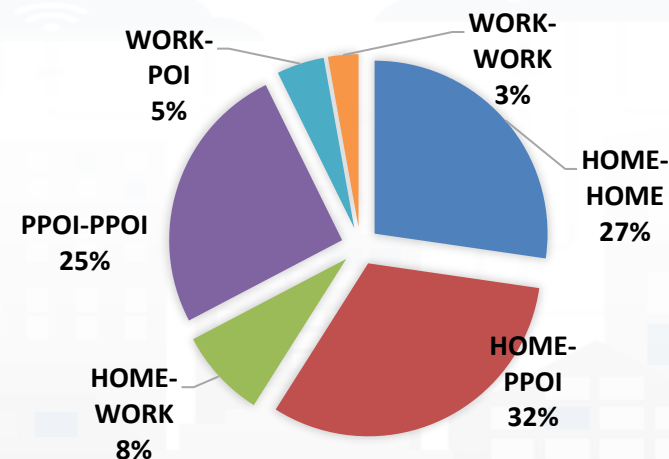
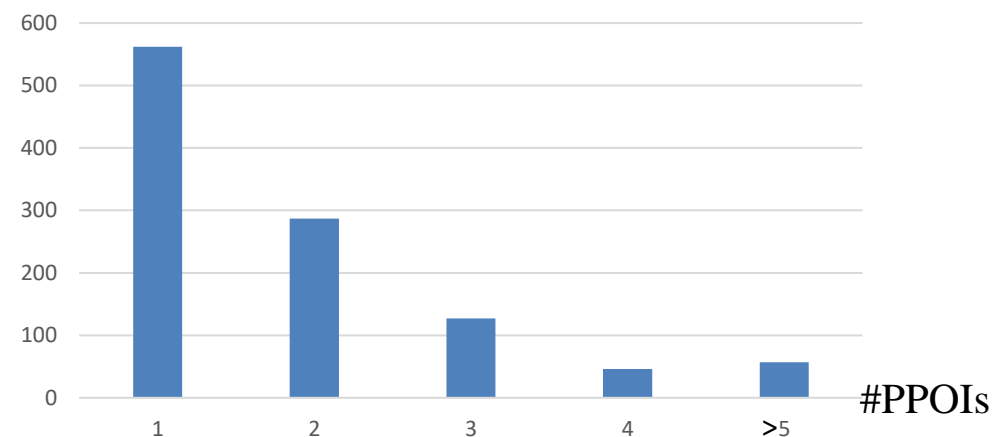
– 130 PPOIs are feedbacked

– 460 survey responses

From 1° August 2017

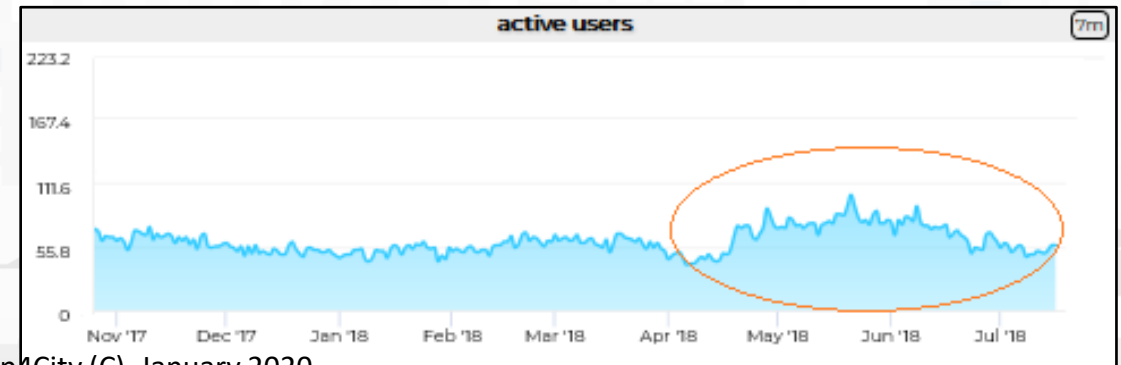
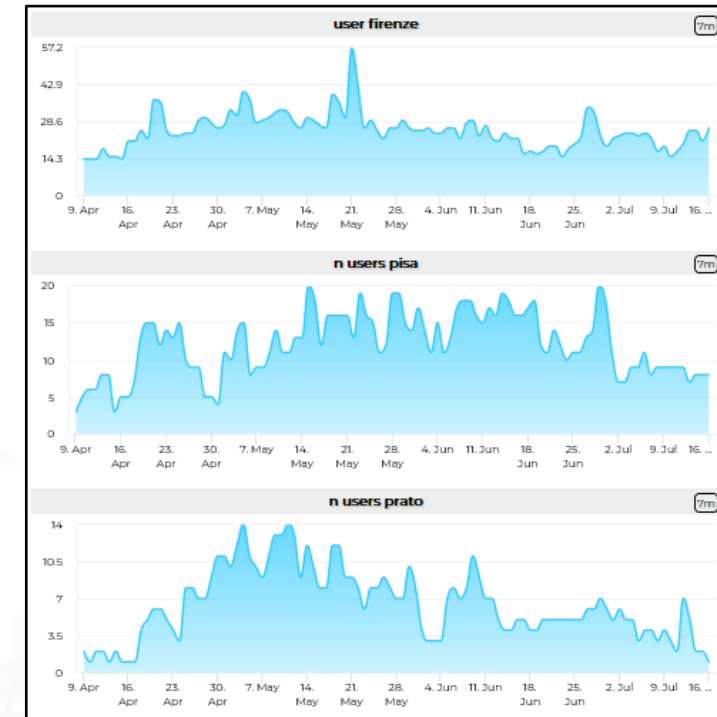
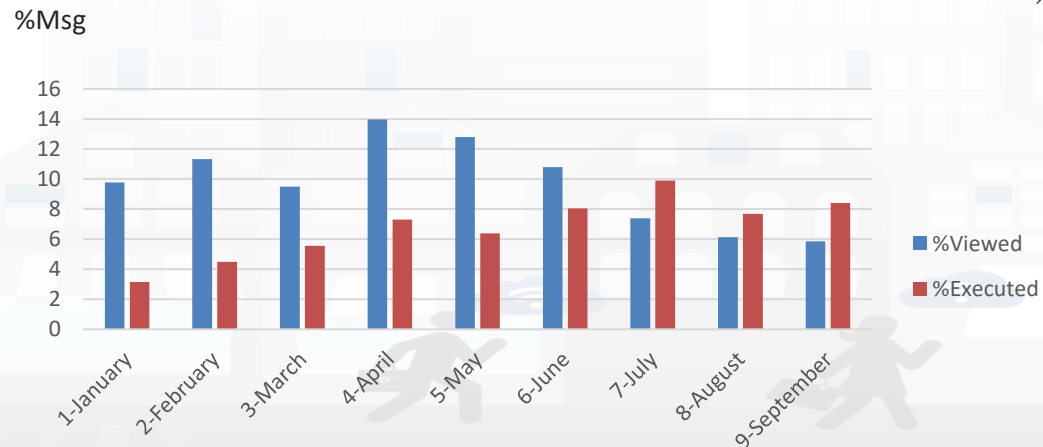
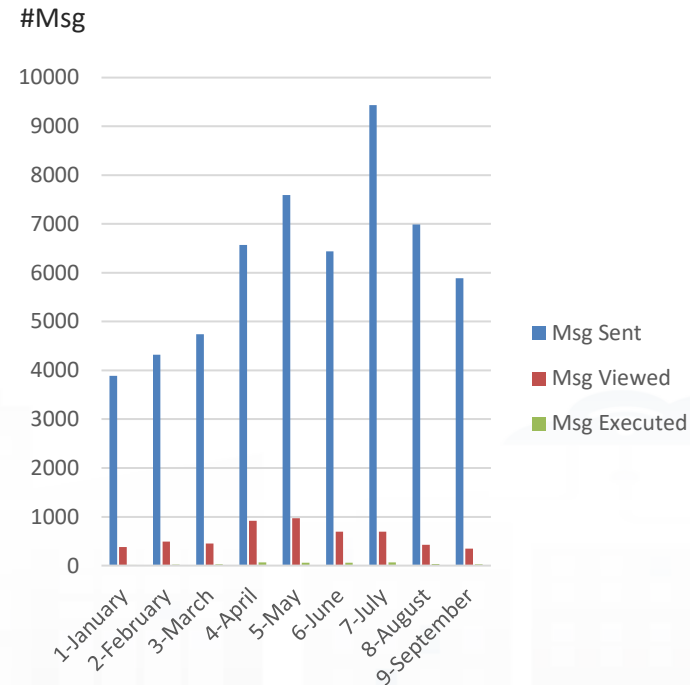
– Built 524 Markov Networks about user's trajectories

Number of users with #PPOIs



Validation of user Engagement

Months	Msg Sent	Msg Viewed	Msg Executed
1-January	3888	380	12
2-February	4319	489	22
3-March	4739	450	25
4-April	6567	918	67
5-May	7594	972	61
6-June	6437	695	55
7-July	9432	697	69
8-August	6988	429	73
9-September	5885	345	49
Total	55849	5375	433



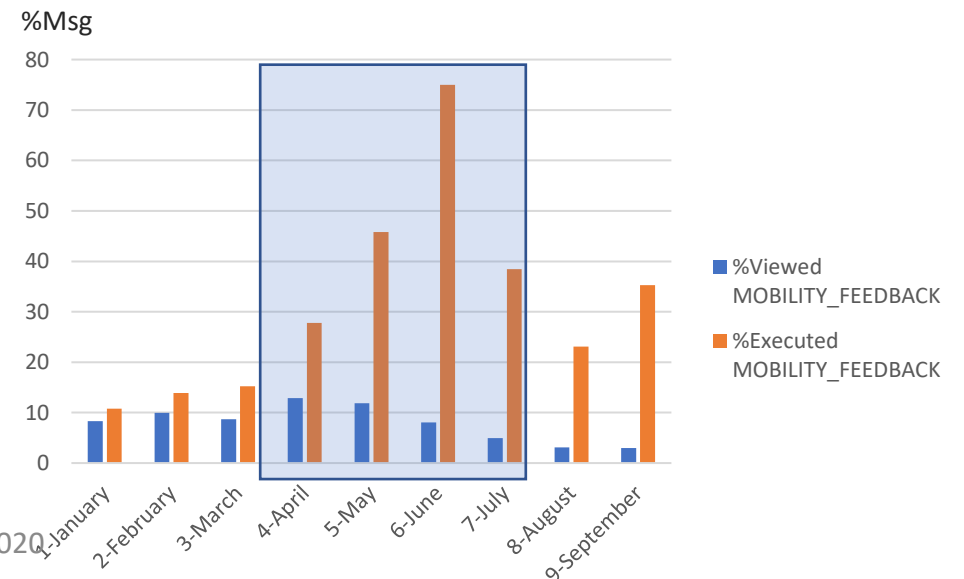
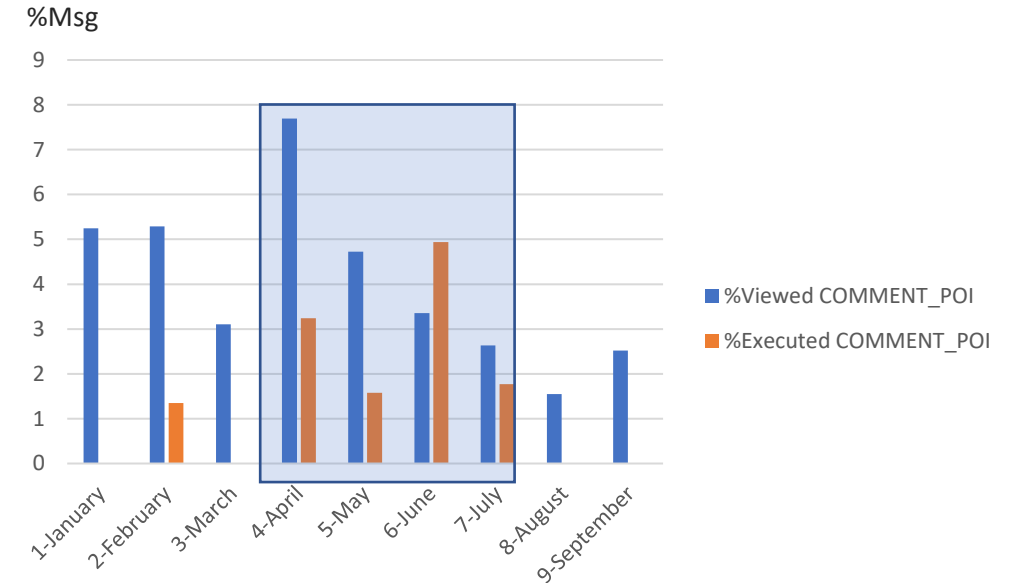
User Behaviour Analysis



VALIDATION

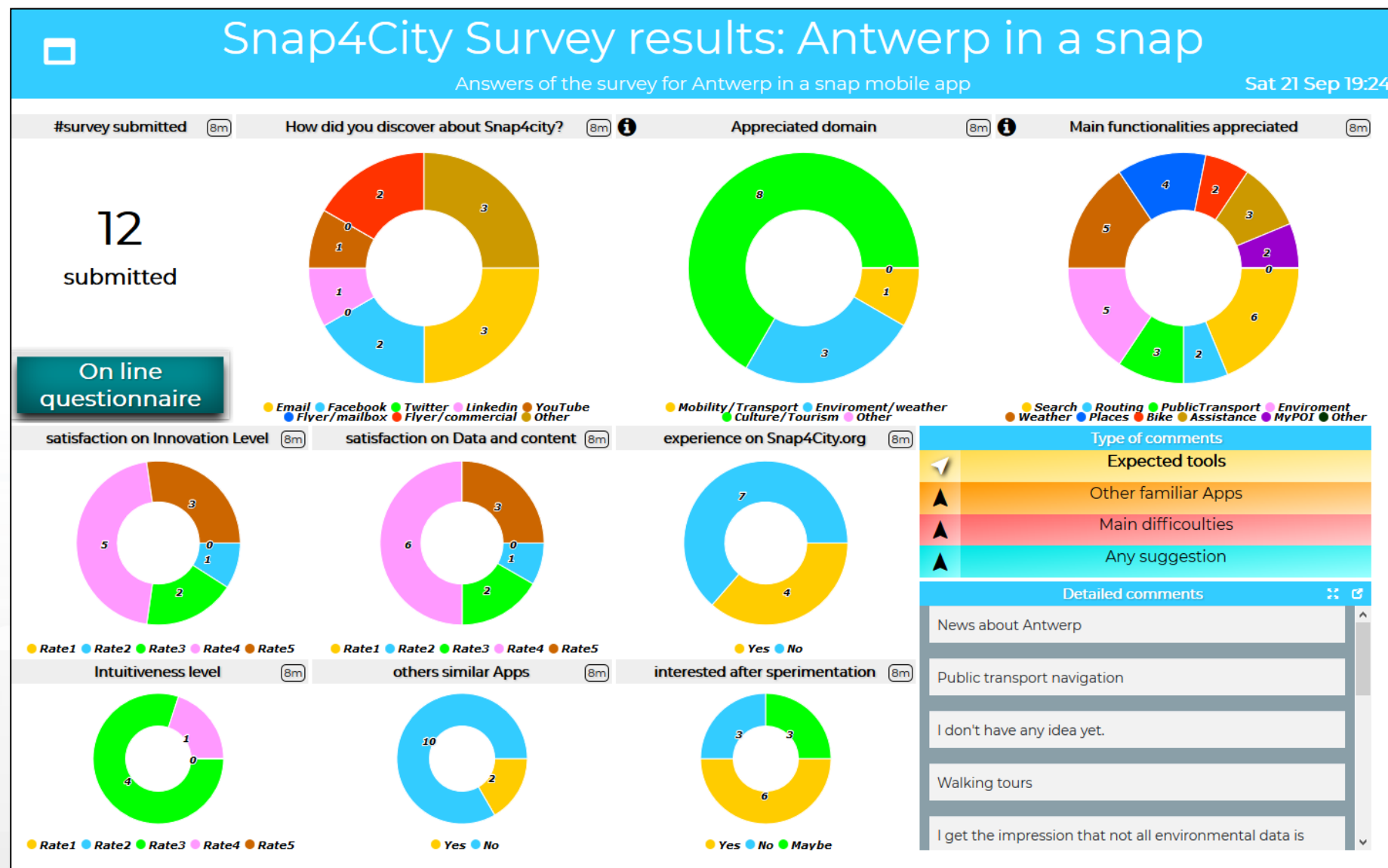
- During the PILOT new rules has been added (30 on a total of 80) and mostly all of them are still online
- COMMENT_POI: requires more user interaction and not very contextualized (POI proximity) → higher rate of sent, lower rate on execution
- MOBILITY_FEEDBACK: requires less user iteration and very contextualized (user in MOBILITY) → normal rate of sent, high rate on execution

	Msg Sent	Msg Viewed	Msg Executed
COMMENT_POI	21632	804	15
MOBILITY_FEEDBACK	5378	371	94



<https://www.snap4city.org/dashboardSmartCity/view/index.php?iddashboard=MTc2OQ==>

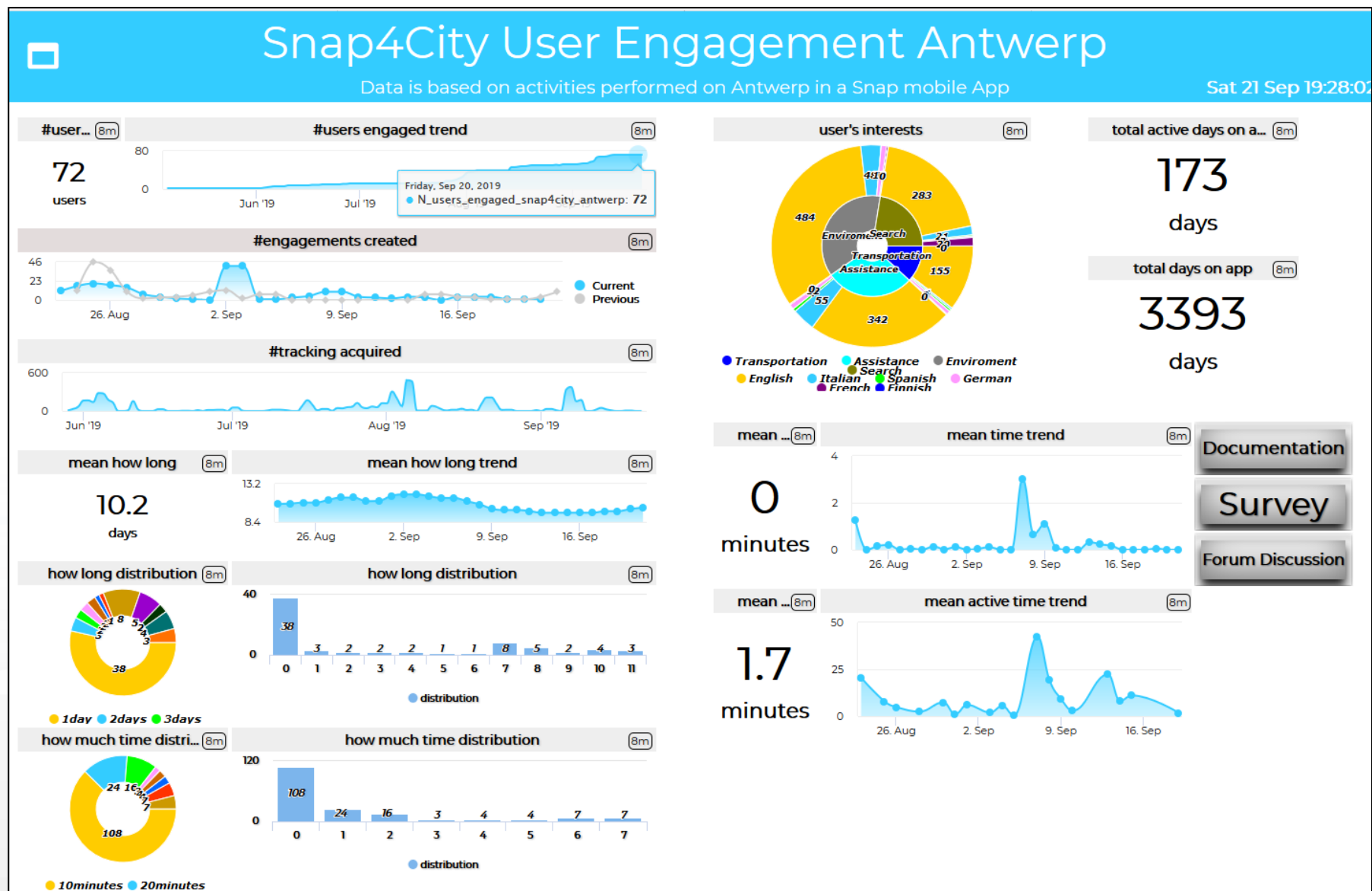
Dashboard
created to monitor
in real time the
answers to the
survey provided
on the Mobile
App directly by the
Engagement tool



<https://www.snap4city.org/dashboardSmartCity/view/index.php?iddasboard=MTc1OQ==>

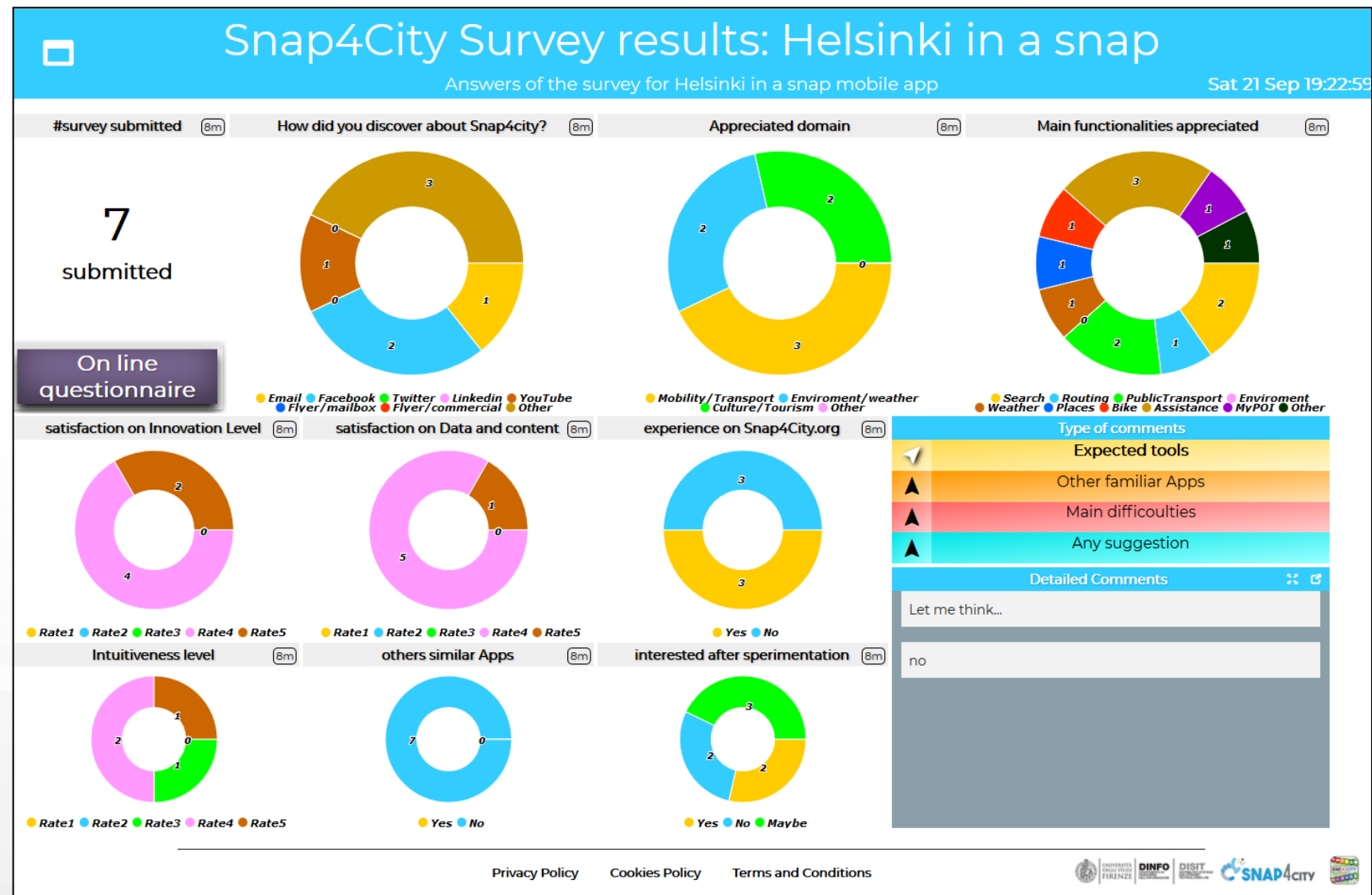
Dashboard monitoring the Mobile App:

- Collecting the clicks
- Describing the community of users in terms of the profile aspects
- Measuring the time spend, and topics of interest of the users, etc.



<https://www.snap4city.org/dashboardSmartCity/view/index.php?iddashboard=MTc2OA==>

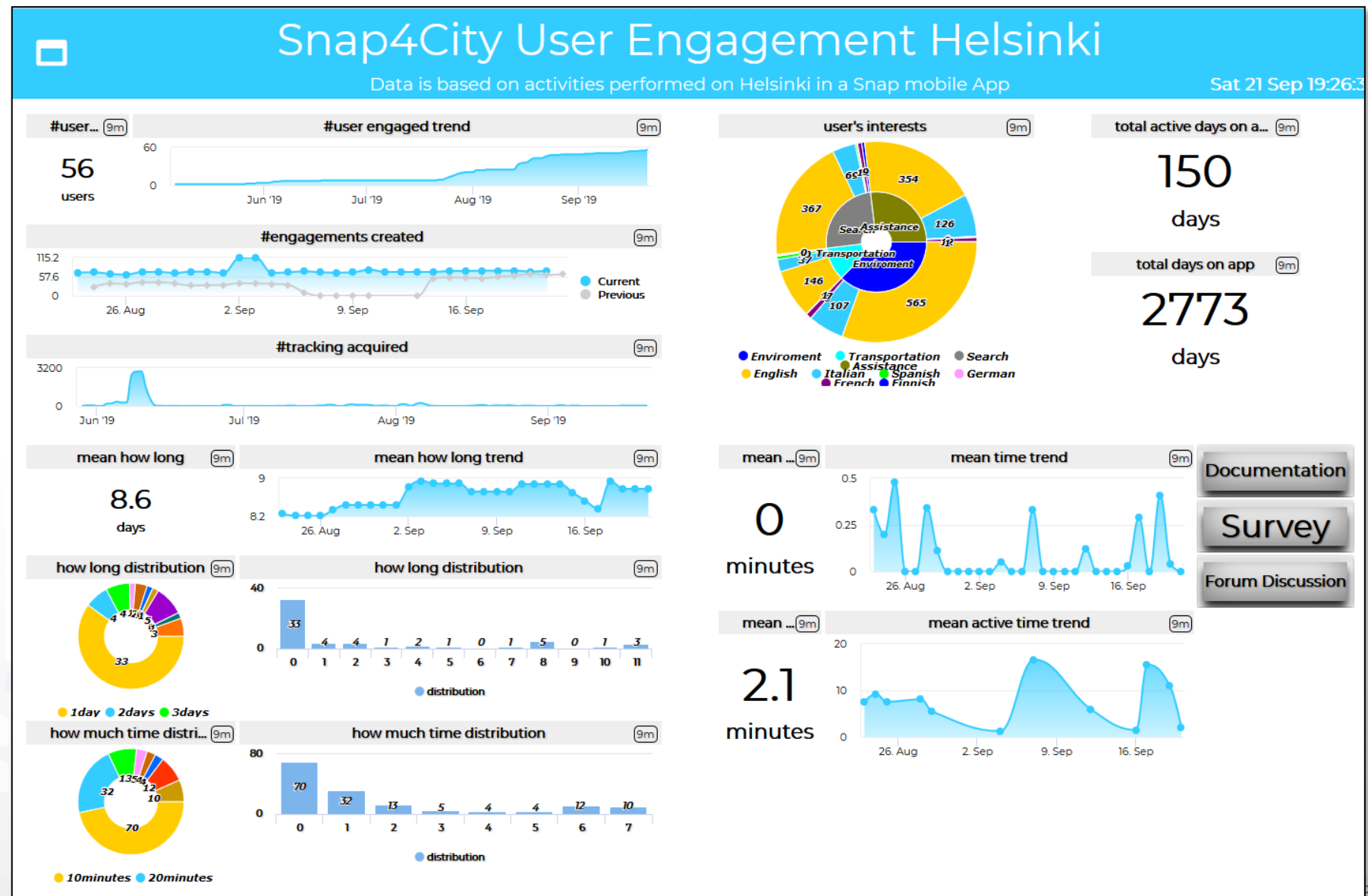
Dashboard
created to monitor
in real time the
answers to the
survey provided
on the Mobile
App directly by the
Engagement tool



<https://www.snap4city.org/dashboardSmartCity/view/index.php?iddashboard=MTc1OA==>

Dashboard monitoring the Mobile App:

- Collecting the clicks
- Describing the community of users in terms of the profile aspects
- Measuring the time spend, and topics of interest of the users, etc.



TOP

Connected Drive

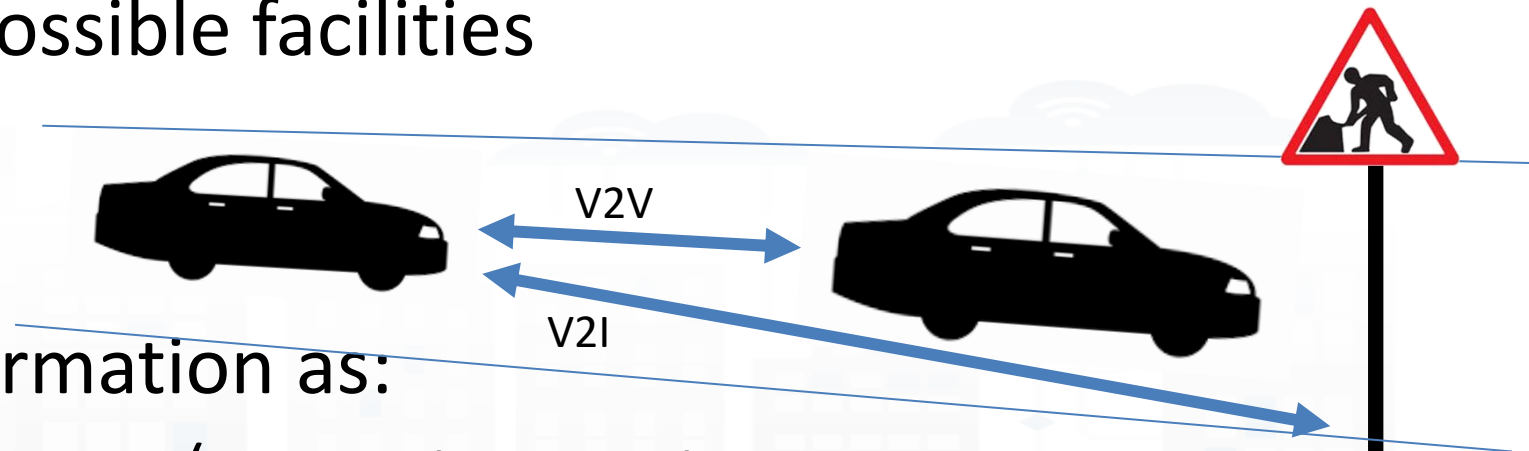


Main Concept of Connected Drive

- Different kinds of communications may arrive on the vehicles on board devices
- Mobile Phones can be a possible facilities

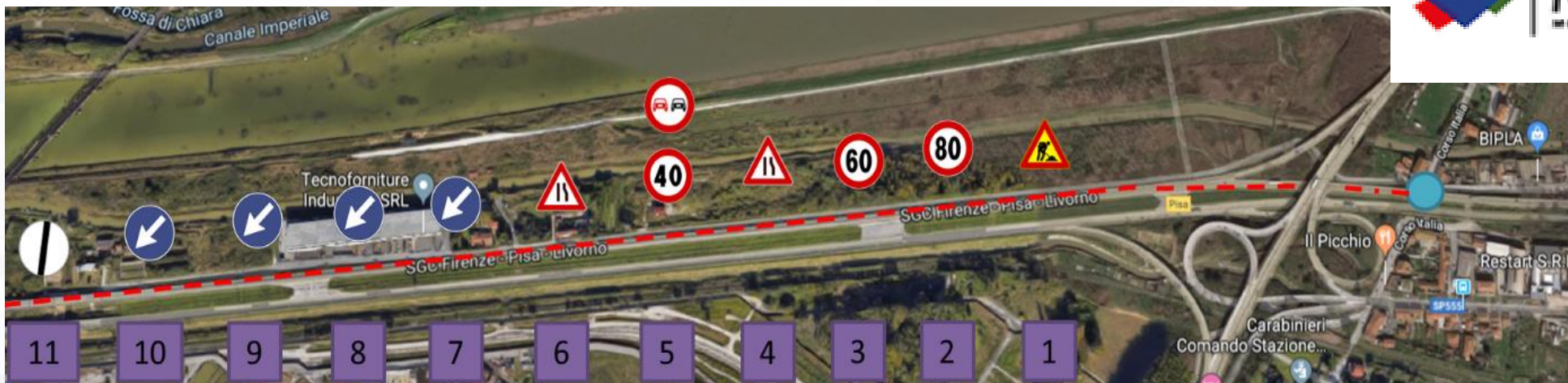


- Geolocated Real time Information as:
 - Alerting, dynamic digital signage (may not present physically on the road)
 - Supporting autonomous driving vehicles



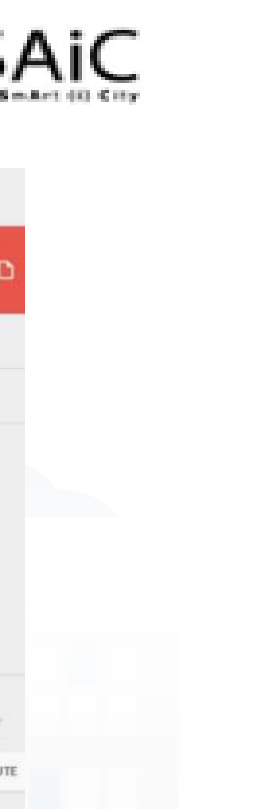
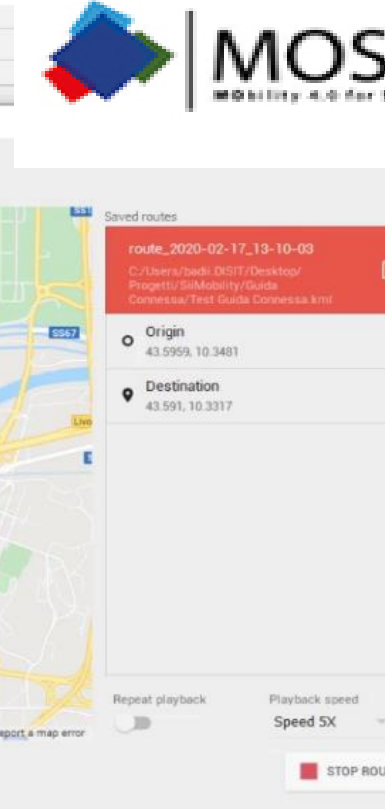
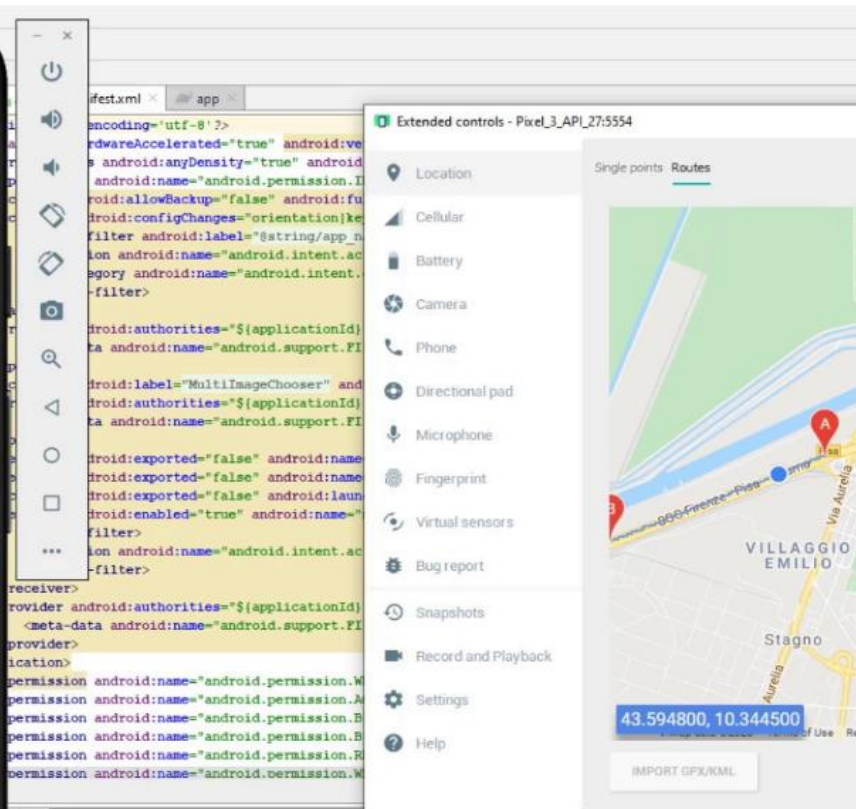
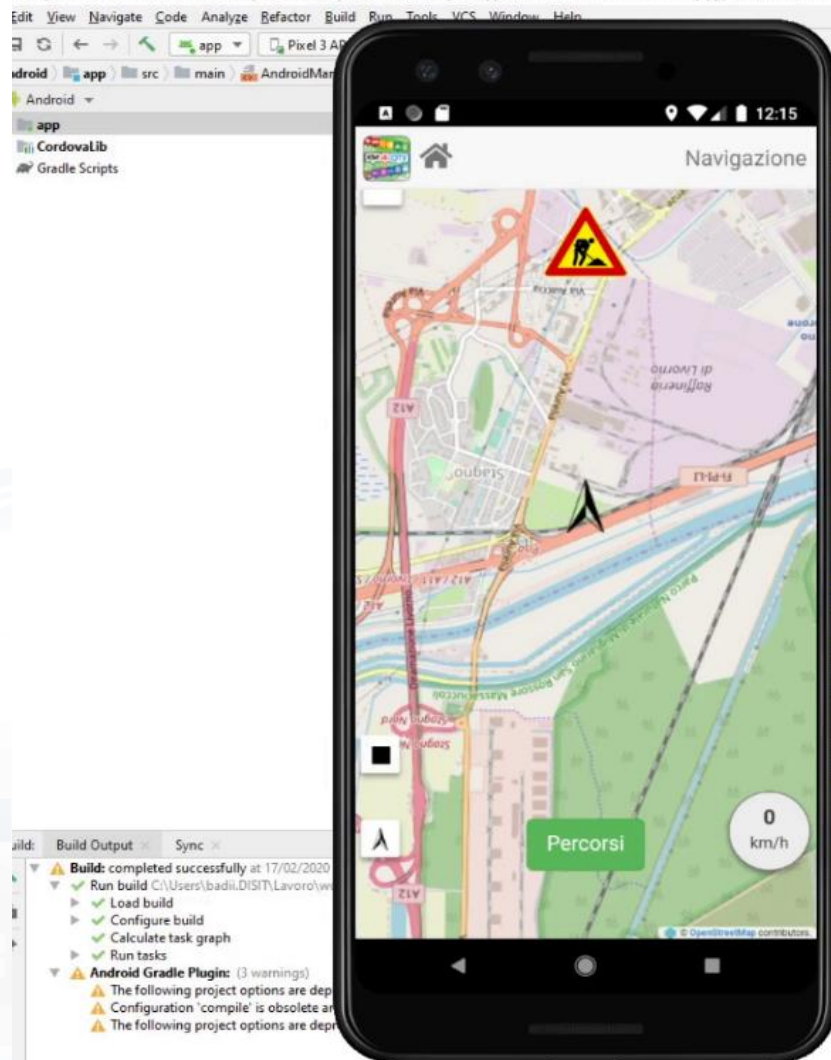
Experimentation on «Toscana Dove Cosa»

- Mobile App supporting connected Drive V2I connections:
 - <https://play.google.com/store/apps/details?id=org.disit.toscana&hl=it>
 - <https://apps.apple.com/it/app/toscana-where-what-km4city/id1064554200>
 - For the MOSAiC project and pilot in Tuscany
- The mobile App has a Navigator which includes now the acquisition of connected drive messages



Scenario

Android Studio interface showing the project structure and the AndroidManifest.xml file.



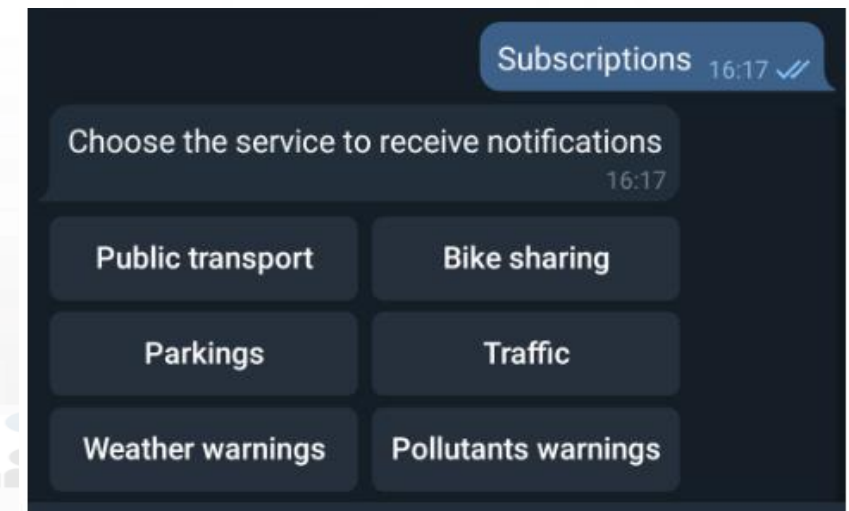
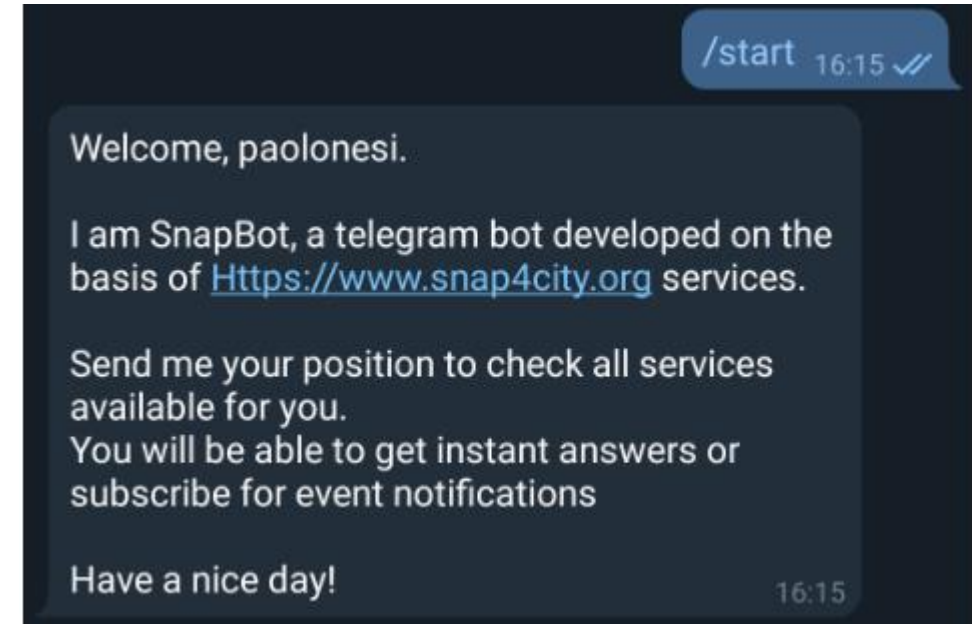
TOP



Integration with Telegram: SnapBot solution



- provides real time smart city services to **Telegram** users, geolocalized, when you like, what you like
- active on Tuscany in all provinces and cities according to the data accessible on <https://www.snap4city.org>
- Services on
 - Public Transport (more than 10 different operators),
 - bike sharing, parking lots,
 - traffic flow, weather warnings,
 - Air quality, pollutant,
 - find your location, etc.





Tap on the hour you prefer to receive 3 notification everyday for the Bike Sharing service 16:18

00:00	01:00	02:00	03:00	04:00	05:00
06:00	07:00	08:00	09:00	10:00	11:00
12:00	13:00	14:00	15:00	16:00	17:00
18:00	19:00	20:00	21:00	22:00	23:00

Qualità dell'aria 02:22 ✓

Qualità dell'aria rilevata dal sensore più vicino alla posizione:

- Temperatura: 8.10 °C
- Umidità: 97.50%
- CO: 0.3 µg/m³
- CO₂: 499.0 µg/m³
- NO: NaN µg/m³
- NO₂: 56.1 µg/m³
- O₃: 20.9 µg/m³
- PM₁₀: 13.8 µg/m³
- PM_{2.5}: 12.2 µg/m³

Public transport 16:41 ✓

Choose a bus stop: 16:42

Giorgini	Giorgini
Vittorio Emanuele	Montelatici

Giorgini - FM0256

- 17:12 - [55] → Cappuccini
- 17:29 - [55] → Cappuccini
- 17:45 - [55] → Cappuccini
- 18:01 - [55] → Cappuccini
- 18:17 - [55] → Cappuccini
- 18:33 - [55] → Cappuccini

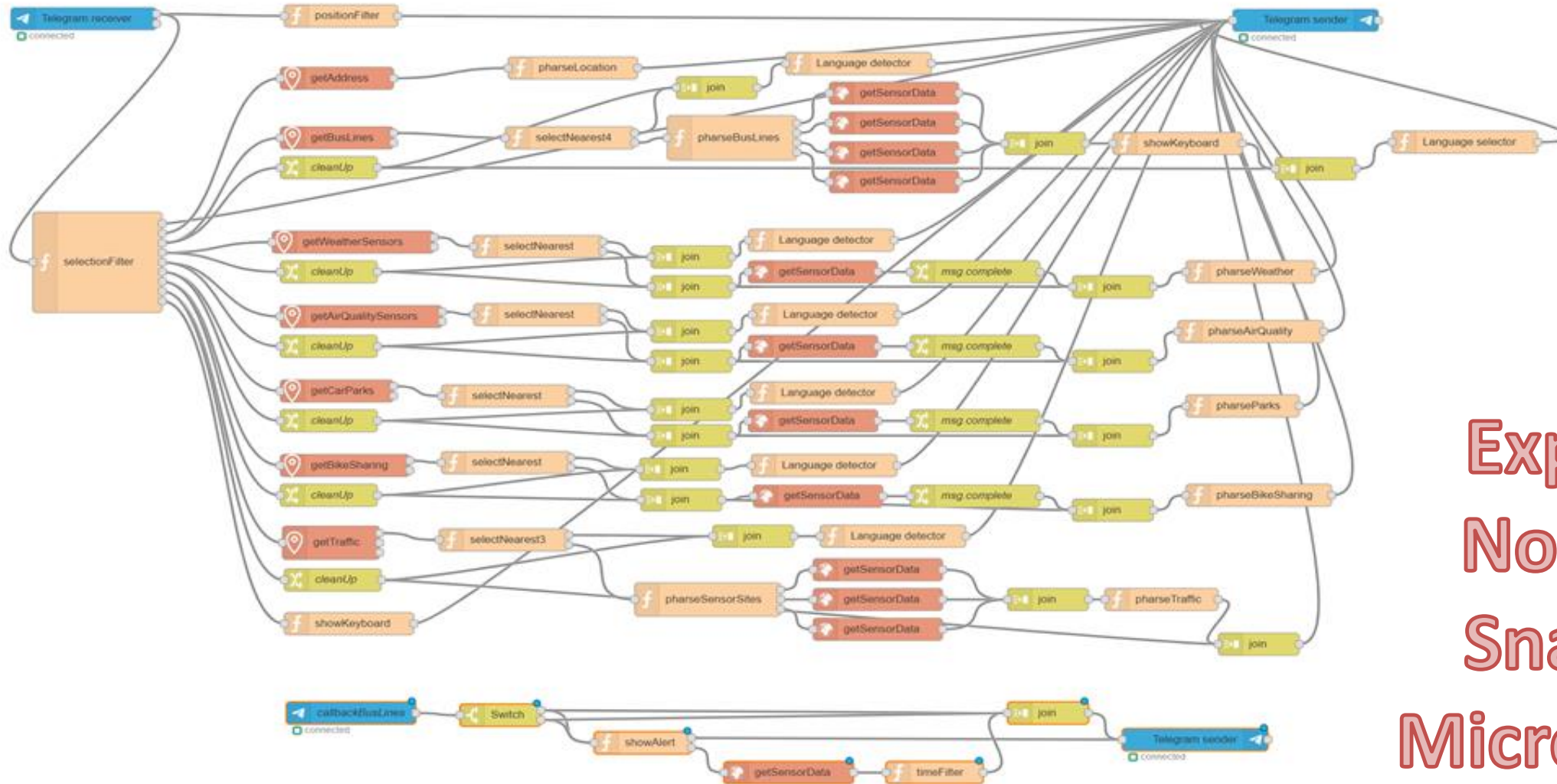
16:43

Trasporti pubblici 14:53 ✓

Ho trovato 6 linee vicino a te:

- 24 - ATAF&LINEA
Grassina → Bagno A Ripoli Robinson
- 49 - ATAF&LINEA
Grassina 02 → Bagno A Ripoli Robinson
- 48 - ATAF&LINEA
Il Roseto 01 → Bagno A Ripoli Robinson

IOT App of SnapBot: OneShot Services



Exploiting
Node-RED
Snap4City
MicroServices

TOP

ADVANCED SMART CITY API, MICROSERVICES, SNAP4CITY API

FROM CITY
DASHBOARD TO
APPLICATIONS

DATA GATHERING
AND CITY DATA
KNOWLEDGE
MANAGEMENT

IoT EDGE DEVICES
AND NETWORKS

IoT APPLICATIONS
VS IoT EDGE
DEVICES

ADVANCED
SMART CITY API
MICROSERVICES
SNAP4CITY API

SNAP4CITY
LIVING LAB FOR
COLLABORATIVE
WORK

DATA ANALYTICS
BUSINESS
INTELLIGENCE,
WHAT-IF AND
SIMULATION

SNAP4CITY
ANALYTICS AND
ECOSYSTEM. OPENED
TO DEVELOPERS
AND STAKEHOLDERS

TWITTER
VIGILANCE: SOCIAL
ANALYTICS

HOW TO ADOPT
SNAP4CITY, AND
OUR ROADMAP

DECISION SUPPORT
SYSTEM AND CITY
RESILIENCE

SNAP4CITY
AND KM4CITY
PROJECTS

SNAP4CITY THE
VIEW OF THE
ADMINISTRATORS



Some structures from Km4City model

ServiceMap: <https://servicemap.km4city.org>

The screenshot displays the ServiceMap application interface. The main map shows Florence, Italy, with various service overlays including bus stops, green areas, and bike lanes. Several information panels are visible:

- Top Left Panel:** Search and selection tools. It includes a "Fermate Firenze" tab, a "Comuni in Toscana" tab, and a "Ricerca Testuale" field. Below these are dropdown menus for "Seleziona una provincia:" (FIRENZE) and "Seleziona un comune:" (FIRENZE). The "Actual Selection" shows "Servizio: PERGOLA".
- Top Center Panel:** Information for "Giardino di piazza dell'Indipendenza". It includes a "LINKED OPEN GRAPH" button, a "Tipologia: Entertainment - Green_areas" label, a "Digital Location" label, an "Indirizzo: PIAZZA DELLA INDIPENDENZA, 15" address, a "Cap: 50129" postal code, a "City: FIRENZE" label, a "Prov.: FI" province label, and a "Note: areeverdi238" note. A "Rimuovi dalla Mappa" button is at the bottom.
- Bottom Left Panel:** Weather forecast for Florence. It shows a 5-day forecast from Tuesday to Saturday with icons and temperature ranges. The "Ultimo Aggiornamento: 2015-09-15T09:07:00+02:00" is noted. A "LINKED OPEN GRAPH" button is at the bottom.
- Bottom Center Panel:** Information for "FERMATA : T1 ALAMANNI". It includes a "LINKED OPEN GRAPH" button, a "Linee:" section with a table of bus lines (2, 28, 52, 54), and a note "Dati Real Time al momento non disponibili".
- Bottom Right Panel:** Information for "FERMATA : PERGOLA". It includes a "LINKED OPEN GRAPH" button, a "Linee:" section with a table of bus lines (14, 19, 23, 31, 6), a "Route:" section with a table of routes (6 A, 6 B, 6 A, 6 B) and their paths (NOVELLI → OSPEDALE TORRE GALLI, OSPEDALE TORRE GALLI → NOVELLI), and a note "Dati Real Time al momento non disponibili".
- Right Panel:** A sidebar with "Servizi Regolari" and "Servizi Trasversali" tabs. It includes a "search text into service" field, a "Categorie Servizi" section with a "De/Select All" button, and a list of service categories (Consulate, Controlled_parking_zone, Cycle_paths, Gardens, Green_areas, Historical_buildings, Library, Literary_cafe, Local_health_authority, Monument_location, Museum). Below this is a "Fresh Place" section with "Road Sensors" and "Bus Stops" categories. A "N. risultati for each:" dropdown is set to "Nessun Limite". A "Raggio ricerca" dropdown is set to "area visibile". At the bottom, it shows "Risultati della ricerca" with "Bus Stops: 21 - Linea Bus: 25" and "Direction: LA PIRA → PIAN DI SAN BARTOLO".

Areas, Bus lines, bike lanes, tram, RTZ, etc.

<http://www.km4city.org>

What do you want to do?

- Discover the City
- Points of Interest
- Search
- Public transport
- Bus Ticket
- Car Park
- Events
- Suggestions Near You
- We Recommend
- Weather
- Assistant
- Navigator
- Favourites
- Chronology
- Latest Reviews
- Alert Civil Prot.
- Settings
- Vote APP!
- Information
- About Us

KM4CITY

Choose Services

- Accommodation
- Advertising
- Agriculture And Livestock
- Civil And Edil Engineering
- Cultural Activity
- Education And Research
- Emergency
- Entertainment
- Environment
- Financial Service
- Government Office
- Health Care
- Industry And Manufacturing
- Mining And Quarrying
- Shopping And Service
- Tourism Service
- Transfer Service And Renting

Giardino Di Boboli

Tipo: Digital Location

Descrizione: The Prince's way ends in the Giardino di Boboli, near the Grotta del Buontalenti, that is a very masterpiece of the Mannerist architecture and sculpture

Descrizione: Il Percorso del Principe termina nel Giardino di Boboli, nei pressi della Grotta del Buontalenti, vero e proprio capolavoro dell'architettura e della scultura manierista

KM4CITY

Carrier 4:29 PM

Search: ponte

Results: Ponte, Vecchio, Ponte, Vecchio

KM4CITY

Choose

- Accommodation
- Cultural Activity
- Education
- Emergency
- Entertainment
- Environment & Ag
- Financial Service
- ATM
- Bank
- Financial Institut

NETFLIX

Tutta la posta in un unico posto

Calendario

Posta

Sereno 3° 11°

Microsoft Edge

Roma

Anteprima Sk...

Tuneln Radio

Twitter

SODA

DISPONIBILE SU
Google play

Scarica da
App Store

Scarica da
Windows Store

FIRENZE

ESERCITAZIONE MUGNONE 2016

MUGNONE 2016
28 maggio 2016

Esercitazione Mugnone 2016
28 maggio 2016
0800-1300

Regione Toscana

prevede: Firenze (FI) (ZONA: A3)		
RISCHIO	TEMPI	ALLERTA
IDROGEOLOGICO IDRAULICO RETICOLO MINORE	Dalle ore 13.00 di Venerdì 27 maggio 2016 alle ore 18.00 di Venerdì 27 maggio 2016	GIALLO
IDROGEOLOGICO IDRAULICO RETICOLO MINORE	Dalle ore 18.00 di Venerdì 27 maggio 2016 alle ore 12.00 di	ARANCIONE

Suggerimenti

Post

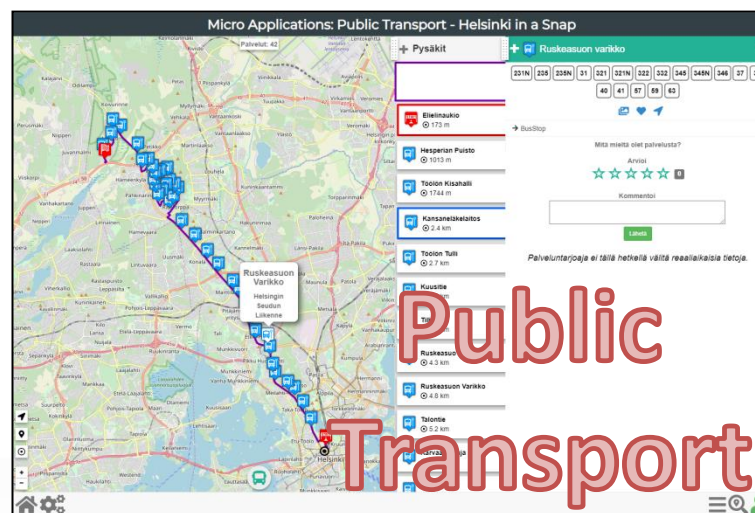
Piazza SS. Annunziata
Tipo: Squares
Distanza: 1949 m
Indirizzo: [Icone]

Piazza Santissima Annunziata
Tipo: Squares
Distanza: 1949 m
Indirizzo: [Icone]

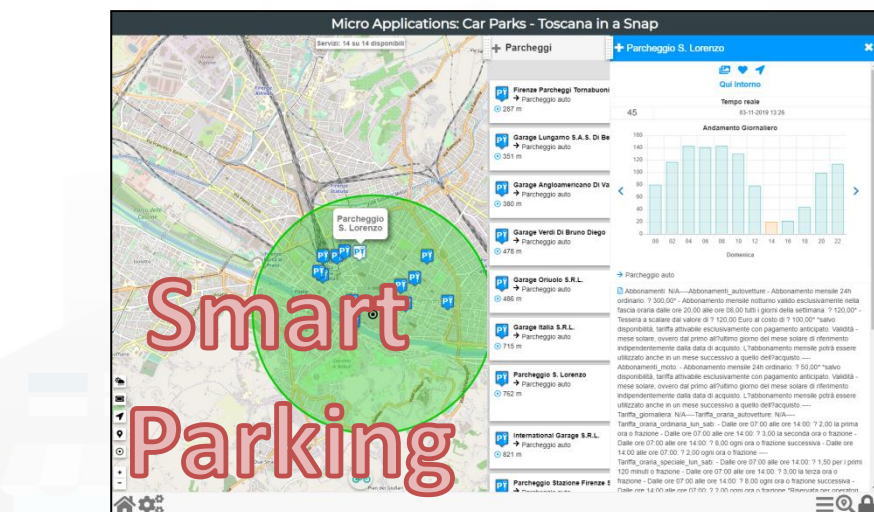
Trattoria Gozzi
Tipo: Trattoria
Distanza: 1975 m

Mostra Tutte le Categorie

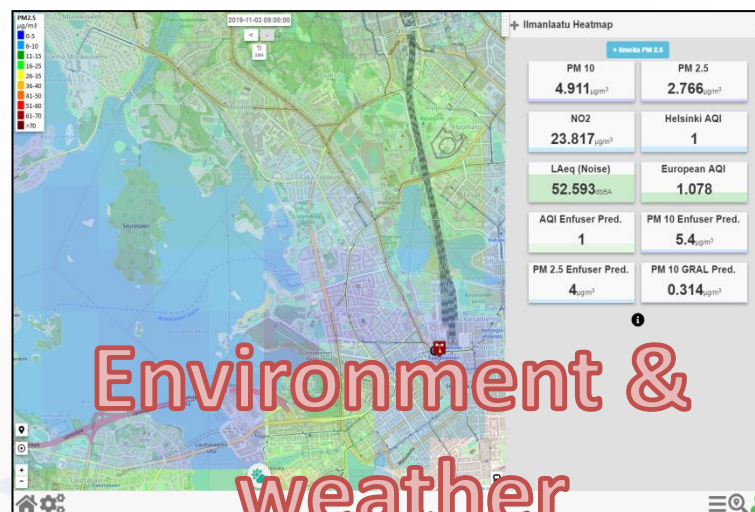
MicroApplications



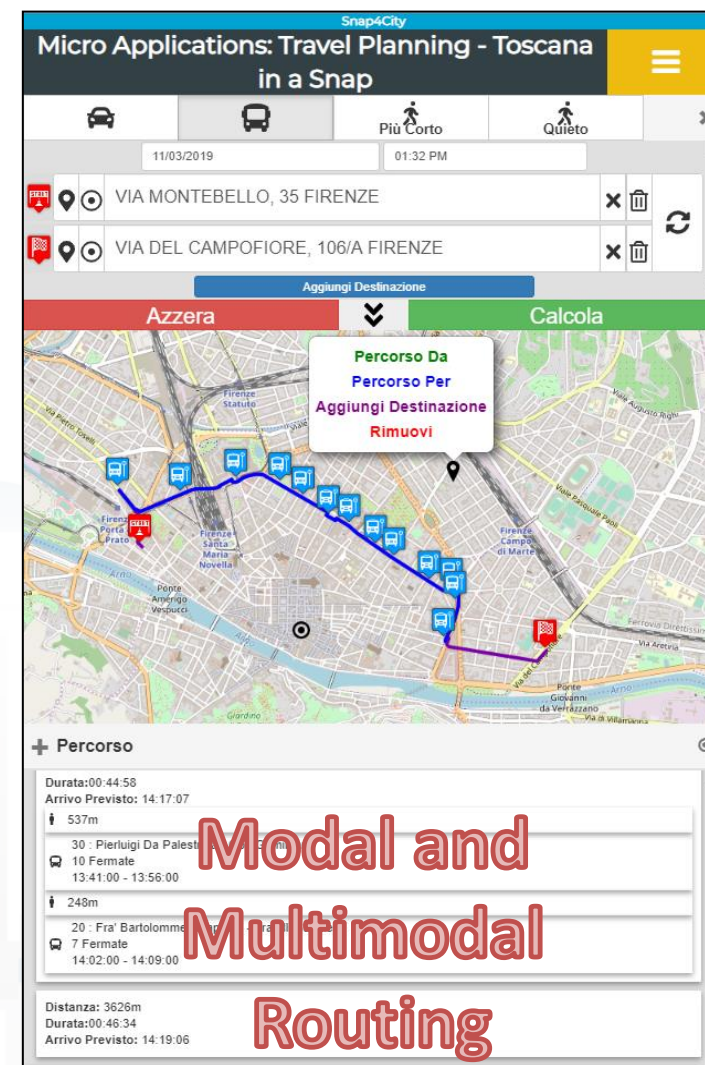
Public
Transport



Smart
Parking



Environment &
weather



Modal and
Multimodal
Routing

Advanced Devevelopment Kit features



- **Exploiting Km4City Advanced Smart City API**
 - Open Source: GitHub
 - Multiplatform: exploiting Apache Cordova Framework
 - Active since 2015
 - Adopted by a community of several Projects, Cities and SME
- **Respecting user privacy:**
 - Anonymous usage vs Authenticated usage (OAuth, email, ...)
- **Modular & Dynamic:**
 - Loading new modules from the WEB, and/or creating App by modular approach
- **Personalization and Profiling:**
 - Personalized menu, proposed POI for search
- **Reaching City Users:**
 - Alerting and notifications by location, by user behaviour

Advanced SmartCity API

Swagger

- Search data: by text, near, along, etc.
 - Resolving text to GPS and formal city nodes model
- Empowering city users: contributions, suggestions, forum discussions, etc.
- Events: Entertainment, critical and mobility
- Public and Private Mobility & Transport, and predictions
- POIs, Cultural and Touristic info
- Health services and predictions
- Environmental information, heatmaps; values
- Profiled Suggestions to City Users
- Traffic flow reconstruction
- Personal Assistant: PAVAL
- User Engagement: goal experiences, and assessment
- *Sharing knowledge among cities → see Knowledge base Management*

The screenshot shows the Swagger UI for the Snap4City API. The left sidebar contains a navigation menu with various system components. The main panel displays the 'Advanced Smart City API' documentation, which includes a list of services and their descriptions. The 'Services' section lists endpoints such as 'Service search near GPS position', 'Service search near a service', 'Service search within a GPS area', 'Service search within a WKT described area', 'Service search within a stored WKT described area', 'Service search by municipality', 'Service search by query id', 'Full text search', and 'Service info'. Each service has a brief description of its functionality. The 'Parameters' section at the bottom shows a table with columns for 'Name' and 'Description', detailing the 'selection' parameter used for searching services.



Snap4City

User: roottooladmin1, Org: DISIT
Role: RootAdmin, Level: 7

Dashboards

My Dashboards

Notificator

IOT Applications

My Personal Data

IOT Directory and Devices

Knowledge and Maps

Micro Applications

External Services

Data Set Manager: Data Gate

Resource Manager: Process Loader

Development Tools

R Studio Development

ETL Development

Knowledge Base Graphs

Knowledge Base Queries

Smart City API Docs: Swagger

Internal API Docs: Swagger

Testing API by Postman

Source Code Access

Management

Settings

User Management and Auditing

Help and Contacts

Documentation and Articles

My Profile

Snap4City portal

Km4City portal

DISIT Lab portal

Smart City API Docs: Swagger

swagger

Select a spec

Advanced Smart City API

Advanced Smart City API

Km4city Web App API

Orion Broker K1-K2 Authentication API

Advanced Smart City API 1.0.0 OAS3

<https://www.km4city.org/swagger/external/ascapi-openapi3.json>

SMART CITY API WEB DOCUMENTATION

Servers

<https://servicemap.disit.org/WebAppGrafo/api/v1>

Services

GET / Service discovery and information

- Service search near GPS position** - It allows to retrieve the set of services that are near a given GPS position. The services can be filtered as belonging to specific categories (e.g. Accommodation, Hotel, Restaurant, etc.), or having specific words in any textual field. It can also be used to find services that have a WKT spatial description that contains a specific GPS position.
- Service search near a service** - It allows to retrieve the set of services that are near a given service identified by its *serviceUri*. The services can be filtered as belonging to specific categories (e.g. Accommodation, Hotel, Restaurant, etc.), or having specific words in any textual field. It can also be used to find services that have a WKT spatial description that contains a specific GPS position.
- Service search within a GPS area** - It allows to retrieve the set of services that are inside a rectangular area. The services can be filtered as belonging to specific categories (e.g. Accommodation, Hotel, Restaurant, etc.), or having specific words in any textual field.
- Service search within a WKT described area** - It allows to retrieve the set of services that are inside a geographic region described using the Well Known Text (WKT) format. The services can be filtered as belonging to specific categories (e.g. Accommodation, Hotel, Restaurant, etc.), or having specific words in any textual field.
- Service search within a stored WKT described area** - It allows to retrieve the set of services that are inside a geographic region described using the Well Known Text (WKT) format, by referring to the WKT with an identifier provided when the WKT is stored. The services can be filtered as belonging to specific categories (e.g. Accommodation, Hotel, Restaurant, etc.), or having specific words in any textual field. The list of available geometries can be retrieved from the [Service Map](#) in the *Search Area* selection box (with *Search Range specific area*). New geometries can be provided using the <http://www.km4city.org/wkt> web service which can store a WKT from a shp file or providing directly the WKT string.
- Service search by municipality** - It allows to retrieve the set of services that are in a specific municipality. The services can be filtered as belonging to specific categories (e.g. Accommodation, Hotel, Restaurant, etc.), or having specific words in any textual field.
- Service search by query id** - It allows to retrieve the set of services associated with a query stored using the [Service Map](#) user interface.
- Full text search** - It allows to retrieve the geolocated entities (not only services) that match with a list of keywords. The results can be possibly filtered to be within a specified distance from a GPS position, or within a rectangular area or inside a WKT geolocated area.
- Service info** - It allows to retrieve information about a service using its *serviceUri*, as an HTML (*format* query parameter set to *html*) or a machine readable JSON document (*format* query parameter set to *json*).

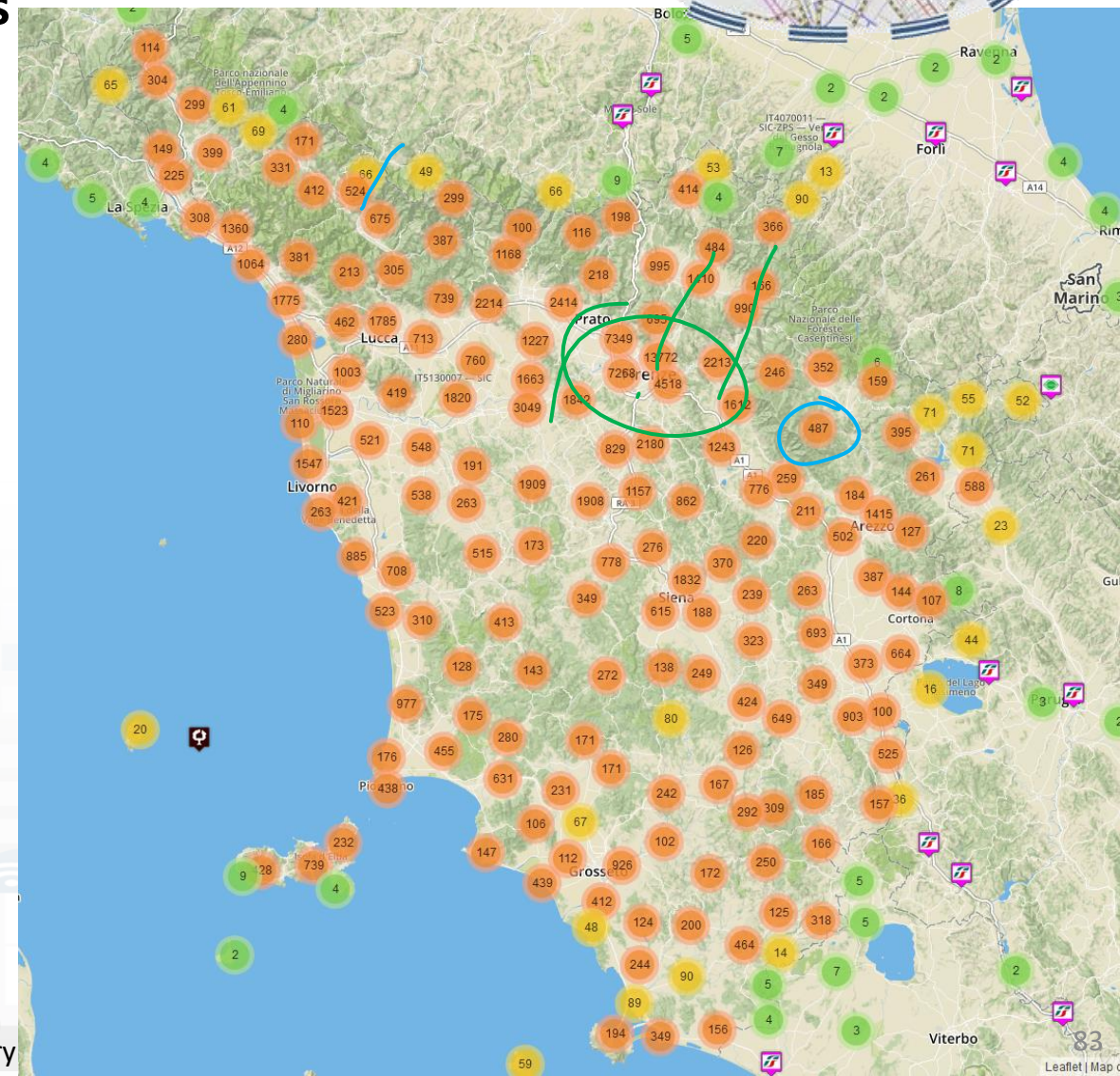
Parameters

Try it out

Name	Description
selection	
string	Through this parameter, the user indicates <i>where</i> the services have to be searched. It could be a boundary within which to search, or a point around which to search.
(query)	Usages & Sample values:

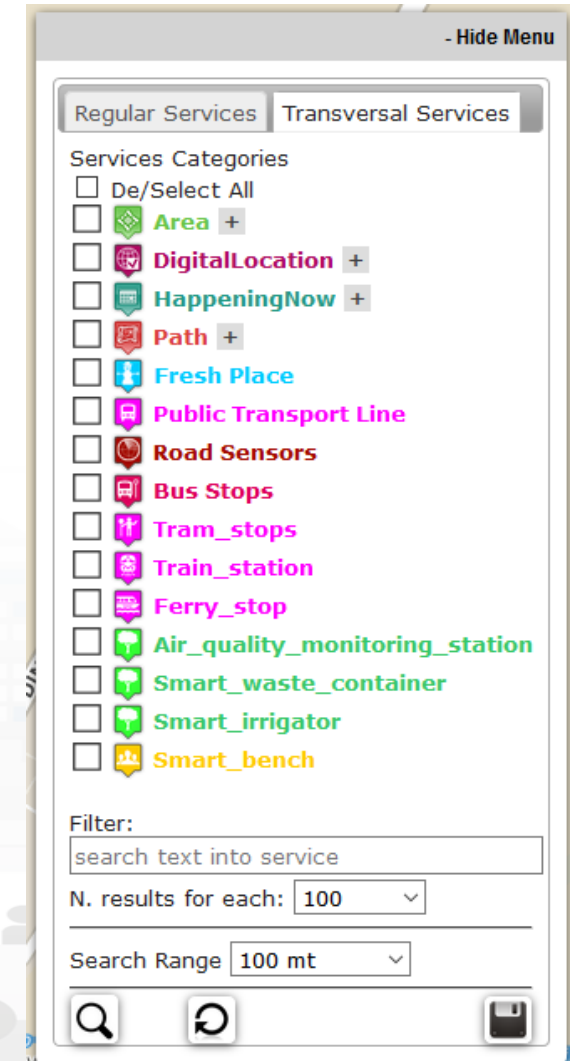
Thematic Data Domain Tuscany

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



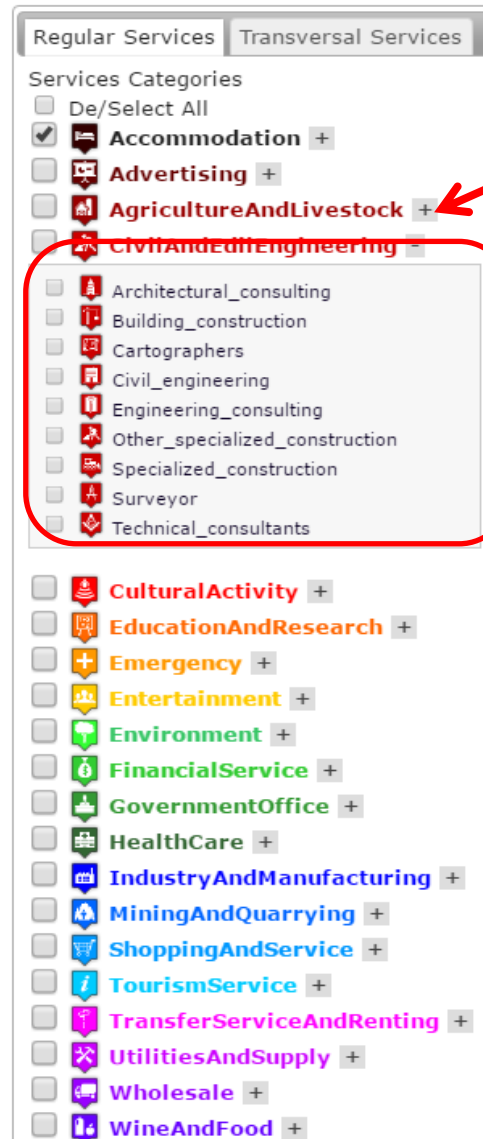
Access to Point of Interest information, POI

- **POI:** point of interest
- **type:** macro and subcategories
- **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



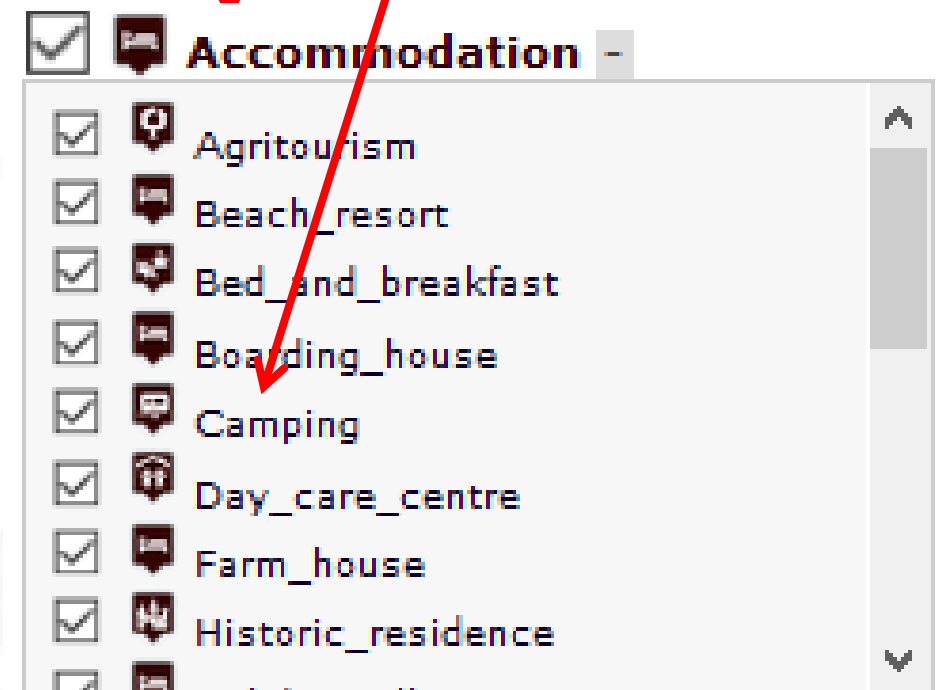
Concepts of Services: Macro and subcategory

A SKOS area into
the Km4City
Ontology and
Knowledge base
for modeling POI
and any element
on map



20 Service Macro Classes (nature)

Service subClasses, (subnature)



Service Information: different kinds of services

AURORA

[LINKED OPEN GRAPH](#)

Tipology: Accommodation - Hotel

Email: info@hotelauroa.info

Website: www.hotelauroa.info

Phone: 055210283

Address: VIA L. ALAMANNI, 5

Cap: 50100

City: FIRENZE

Prov.: FI

TPL STOP : Piazza Stazione (Fr. Cc)

Vaubus

[LINKED OPEN GRAPH](#)

Lines:

FI-LU **FI-VG**

No available routes

Display **50** Bus per page

Search:

Time	Line	Direction
08:48:00 2017-03-20	FI-LU	Piazzale Verdi
08:16:00 2017-03-20	FI-LU	Piazzale Verdi
10:09:00 2017-03-20	FI-LU	Piazzale Verdi
2017-03-20	FI-LU	Piazzale Verdi
2017-03-20	FI-LU	Piazzale Verdi
2017-03-20	FI-LU	Piazzale Verdi

Page 1 of 1

Data currently not available

Loggia San Paolo

[LINKED OPEN GRAPH](#)

Tipology: CulturalActivity - Monument_location

Digital Location

Address: VIA DELLA SCALA, 3

Cap: 50123

City: FIRENZE

Prov.: FI

Photos:

Description: The rounded arches, the stone skeleton and the glazed terracotta medallions recall the model of the Loggiato degli Innocenti. The medallions in glazed terracotta by Andrea della Robbia and his sons Marco and Luca contain seven polychrome figures of Santi Francescani and two works of mercy Cristo conforta un Giovane and Cristo conforta un Anziano. Beneath the portico can be admired the expressive embrace between San Domenico Guzman and San Francesco d Assisi by Andrea della Robbia

Giardino di piazza dell Indipendenza

[LINKED OPEN GRAPH](#)

Tipology: Entertainment - Green_areas

Digital Location

Address: PIAZZA DELLA INDIPENDENZA, 15

Cap: 50129

City: FIRENZE

Prov.: FI

Note: areeverdi238

Remove from map

ZCS_1_D

[LINKED OPEN GRAPH](#)

Tipology: TransferServiceAndRenting - Controlled_parking_zone

Digital Location

Address: VIA GUSCIANA

Cap: 50124

City: FIRENZE

Prov.: FI

Remove from map

General Text Search Features

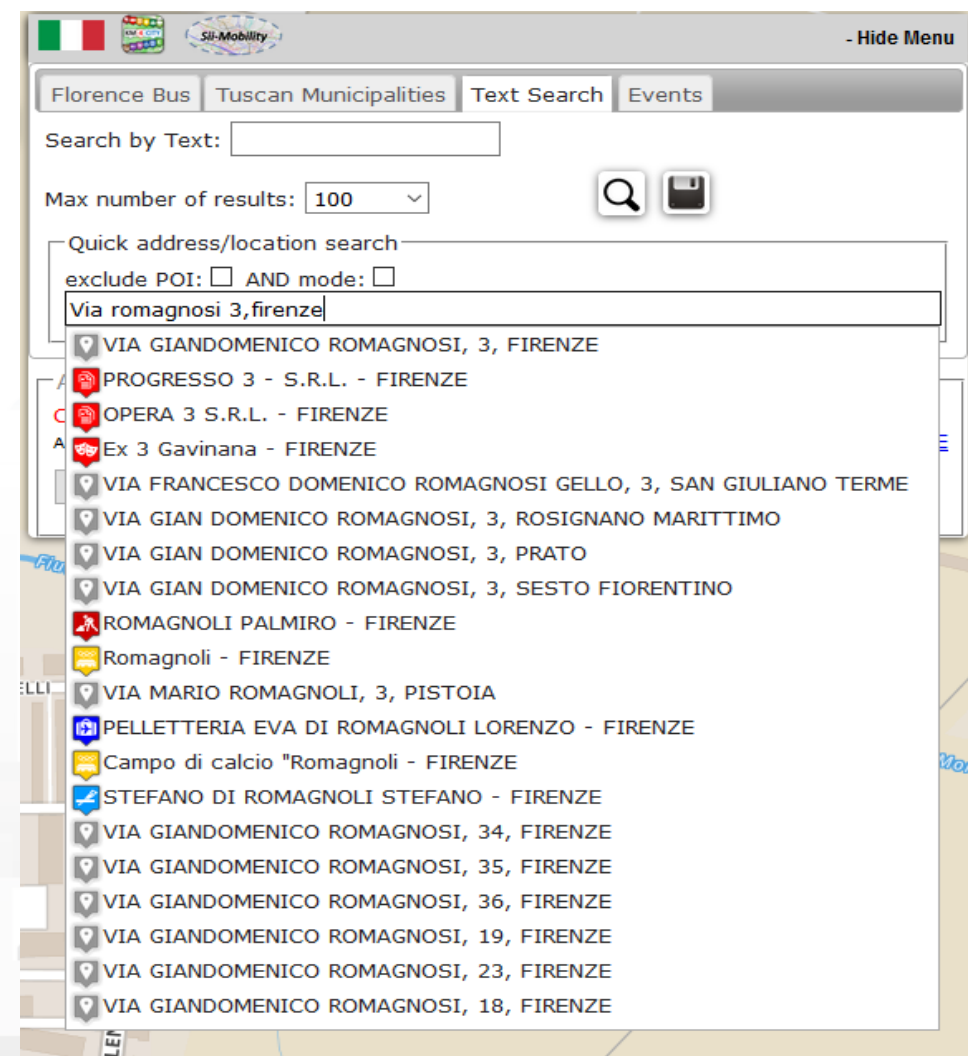
Search by text for POIs via:

- Full text: description, title, macro and category name
- Filtering by macro-cat and subcategory
- Filtering on distance and geometric shape

Search by text with assisted suggestion to get:

- Streets and civic numbers, or POI, locations

Geo resolution, from point to street; from civic to GPS, etc.





Search by Shape and Distance

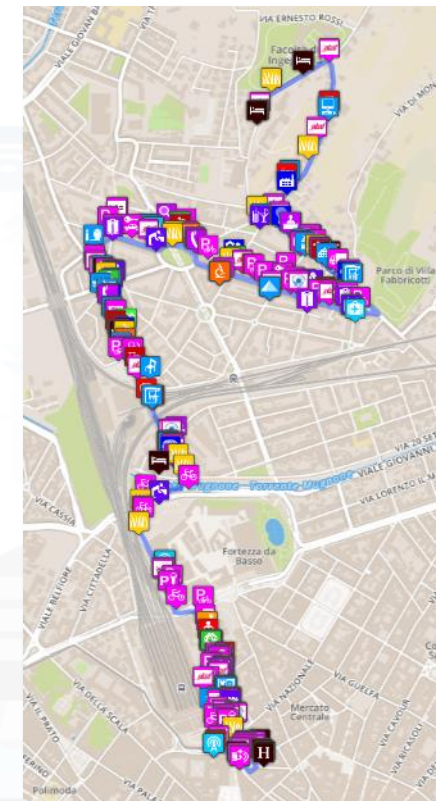
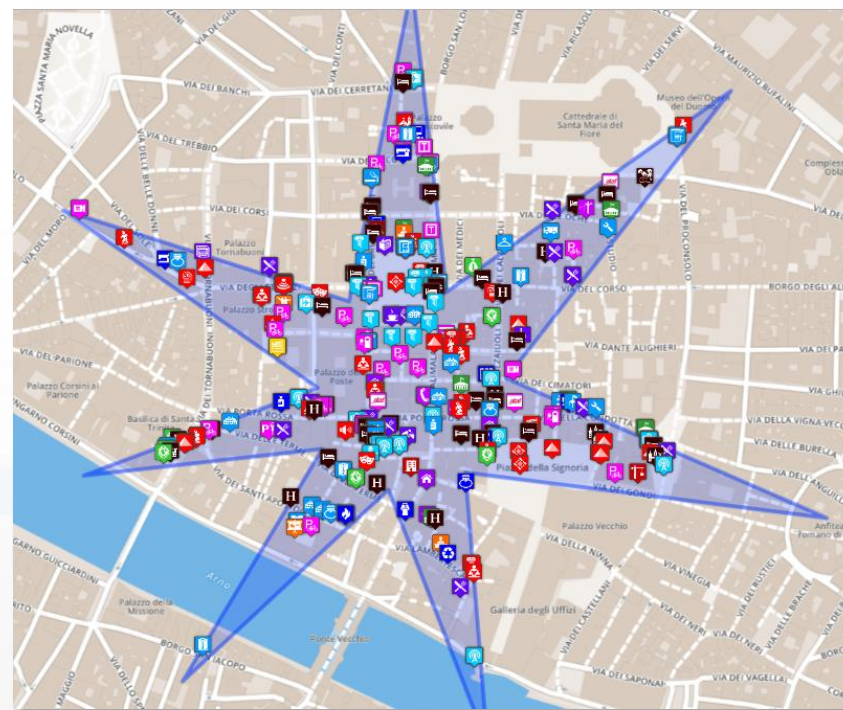
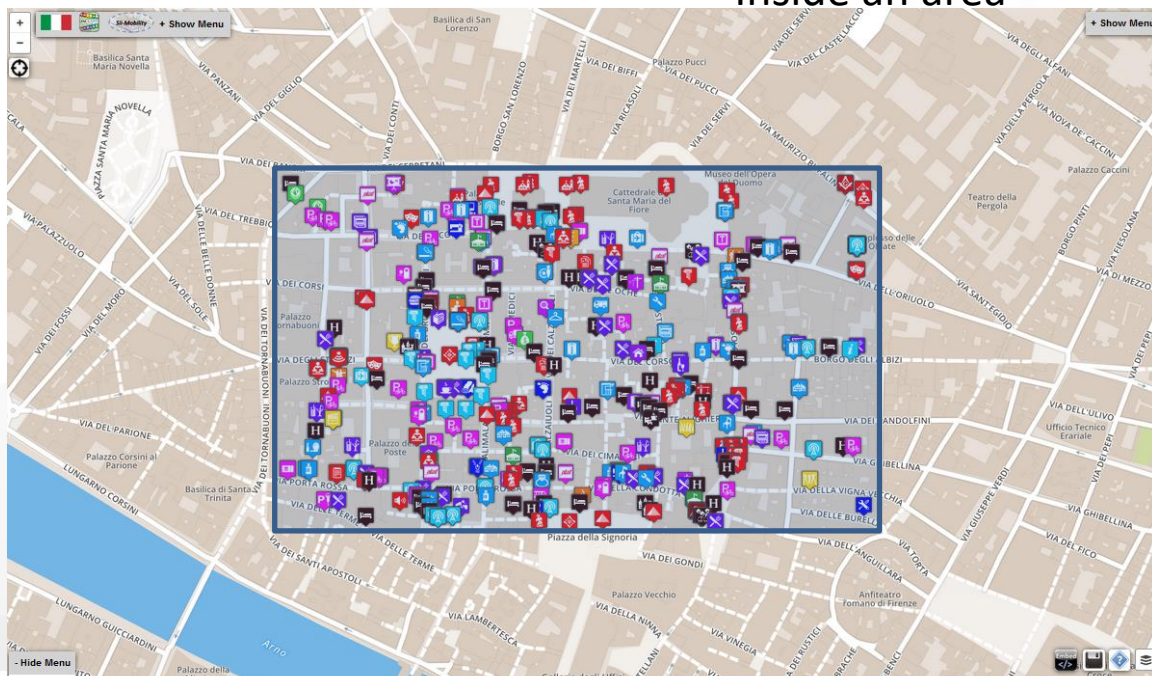
Each request or search in the Km4City model can be referred to a point and a ray, to an area, to a polyline

Inside a closed polyline

Along a polyline



Inside an area



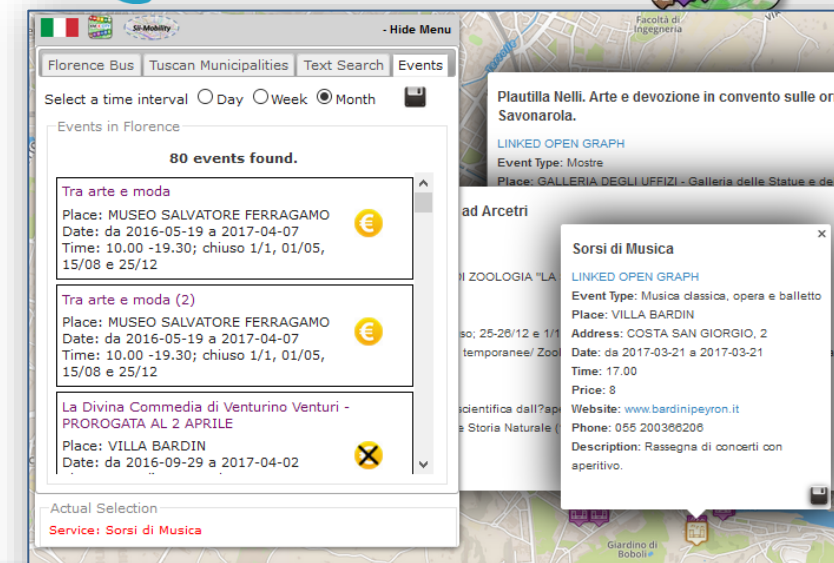
Empowering City Users

- Allow city users to
 - provide comments, images and scores associated with a certain Service (or place, via GPS), discussions on forums, etc.
 - Get list of last contributions of the same kind provided by other users
 - Save favorites
 - Share trajectories,
 - Save and Manage their own data, IOT data, etc.
- Contributions can be:
 - used as feedbacks
 - moderated by a back-office personnel
 - ...
- Connection with powerful servers based on 311 standard it also possible



Access to Event information

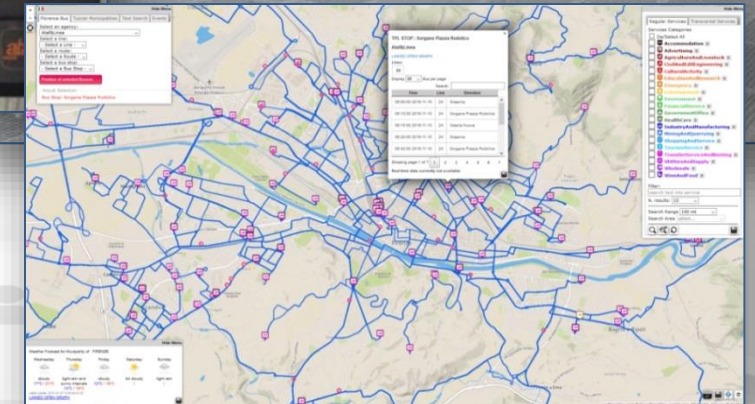
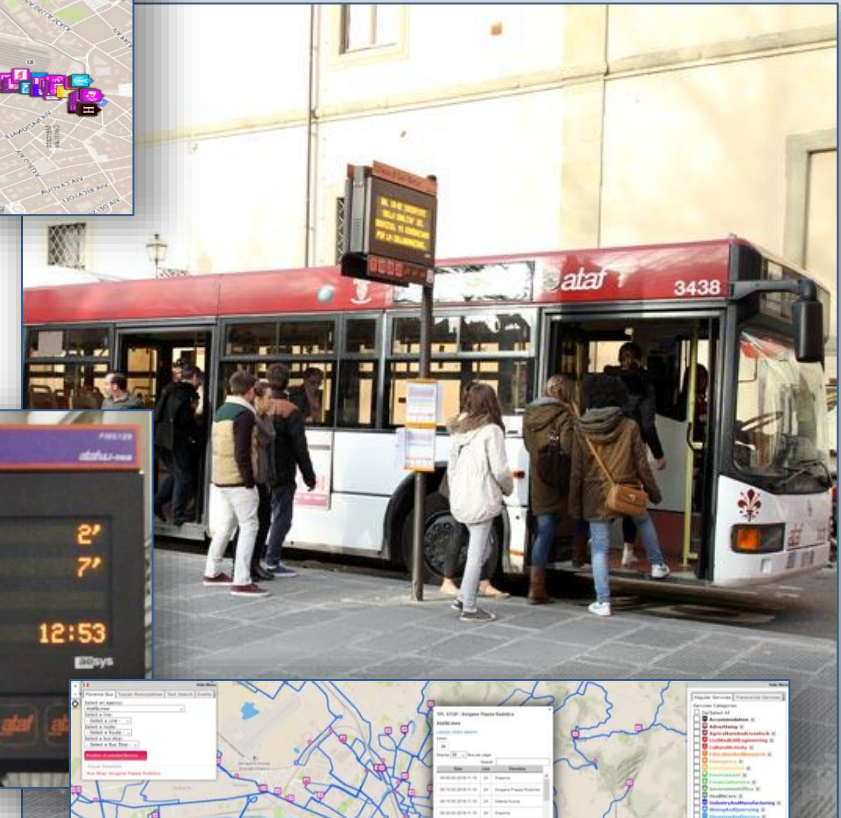
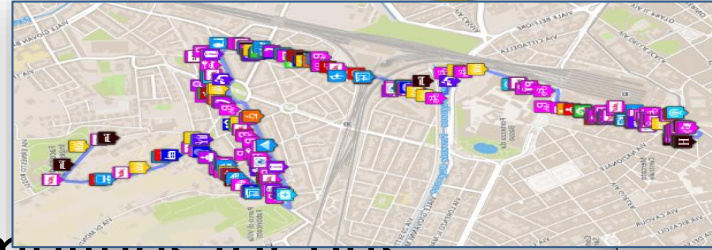
- Getting Traffic Events: ESB, etc.
- Getting Critical Events: CAP standard
- Getting Police events
- Getting Entertainment Events in the city
 - Theater, museum, show, sport, etc.
 - Getting Event details
 - Event kind, and thus ordering
 - in the day, week, and month
 - Location, and thus ordering, or selecting events per area, per residence
 - General information
 - Opening and cost (if any)
 - Etc.



Supporting City Users in using Public Mobility

Public Transportation, PT

- Getting tickets
- Getting bus stops, lines, and timeliness for bus, train and tramline (GTFS, ETL, ...)
- Getting Tunnel and Ferry Status
- Searching Services along a Pub. Transport line or closer to a stop
- Searching the closest bus stops
- searching for BUS stops via name
- real time delays of busses
- Modal/multimodal routing for Pub. Transport
- Tracking fleets, trajectories, etc.
- Get connected drive data

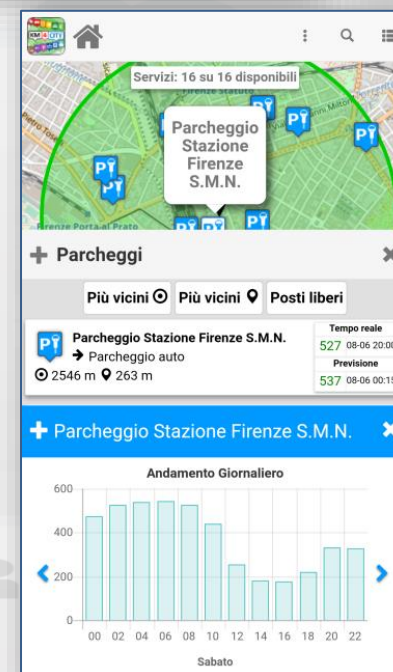
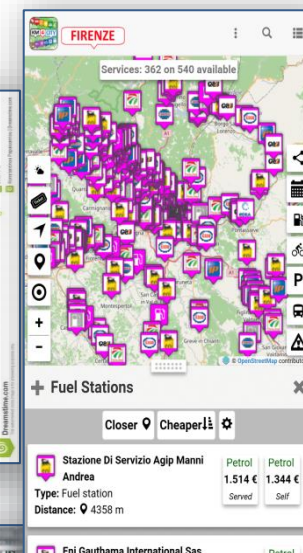




Supporting City Users using Private Mobility

Private Transport

- Parking status (DATEX II, ...)
- Saving car park
- Getting closer parking
- **OBD2 data from your engine or fleet**
- **Getting parking forecast: short and long term**
- Getting closer free space on parking
- Getting **fuel stations** location and fuel product prices
- Getting bike sharing rack status
- Searching Services along a **path** or closer to a point or Service as Hotel, Restaurants, square, etc.
- Getting closer **cycling paths**
- Recharging stations: location and status
- Getting traffic information
- Heatmap where is safer to bike



Private Mobility: routing and navigation paths

To get the path from two points/POIs:

- Shortest for pedestrian
- Quietest for pedestrian
- Shortest for private vehicles
- Multimodal with Public Transportation
- Constrained routing

Search for POIs along the identified Path!

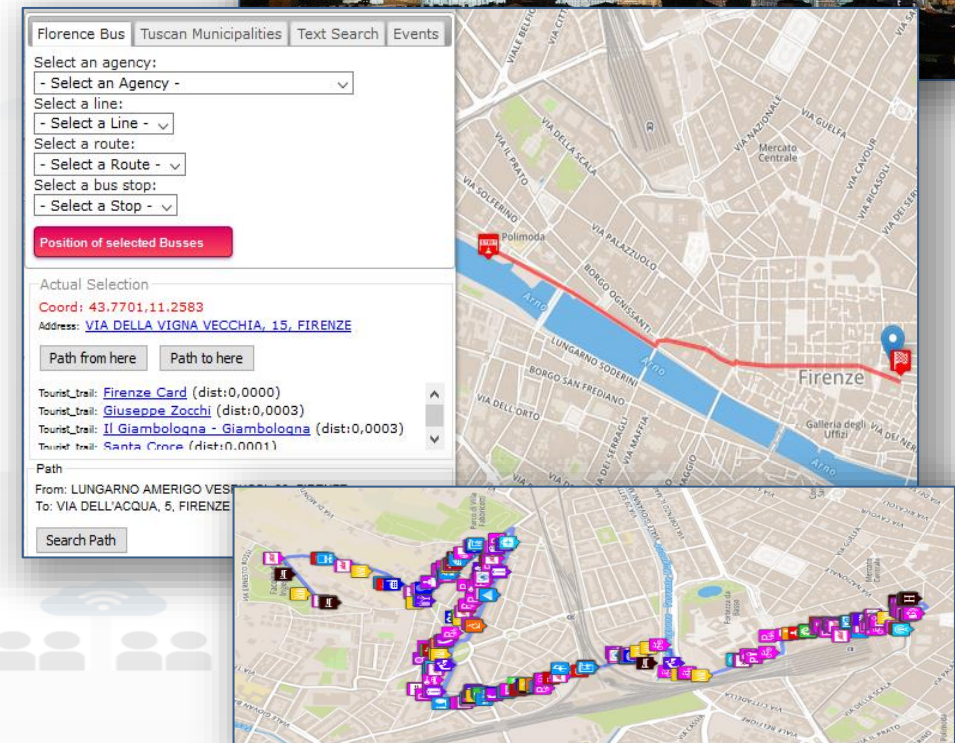
<http://www.disit.org/ServiceMap>

The screenshot displays the Sii-Mobility web application interface. At the top, there are navigation tabs: "Florence Bus", "Tuscan Municipalities", "Text Search", and "Events". Below these, there are dropdown menus for selecting an agency, a line, a route, and a bus stop. A pink button labeled "Position of selected Busses" is visible. A search bar with the text "Quick address/location search" and an "exclude POI" checkbox is present. The "Actual Selection" section shows coordinates (43.7681, 11.2537) and the address "LUNGARNO DEGLI ARCHIBUSIERI, 18, FIRENZE". Below this, there are buttons for "Path from here" and "Path to here". The "Path" section shows the route from "PIAZZA DELLA STAZIONE, 42, FIRENZE" to "LUNGARNO DEGLI ARCHIBUSIERI, 18, FIRENZE" via "foot_shortest". The "Found 1 paths (in 5.959s)" section shows the path length (1262m) and arrival time (12:58:12). A list of path segments is provided, including distances and times for each segment. On the right, a map of Florence shows the route highlighted in red. At the bottom, a zoomed-in map shows the path with various icons representing different points of interest (POIs) along the route.



New Experience to access at Cultural and Touristic info

- Getting location and description of Point of Interests, POIs: culture and tourism first
 - Location, images, phone, URL, etc.
 - Get image, video, audio, ...
- Search for POIs in areas and closer
- Get routing to reach location or POI by walking downtown
 - searching Services along the path
- Search for location, full text assisted
- Leave a score, take a picture, etc.



New way to access at health services

- Searching for pharmacies and hospitals
- Getting the closest hospital first aid locations and status
- Getting real time updated information about the first aid status of major hospitals (triage)



Servizi: 2 su 2 disponibili

corso
enda
dalliera
reggi

Soccorso

Più vicini

Soccorso Azienda Ospedaliera Careggi

Pronto soccorso

Distanza: 1268 m

Triage

5 23 22 3 0

+ Pronto Soccorso Azienda Ospedaliera...

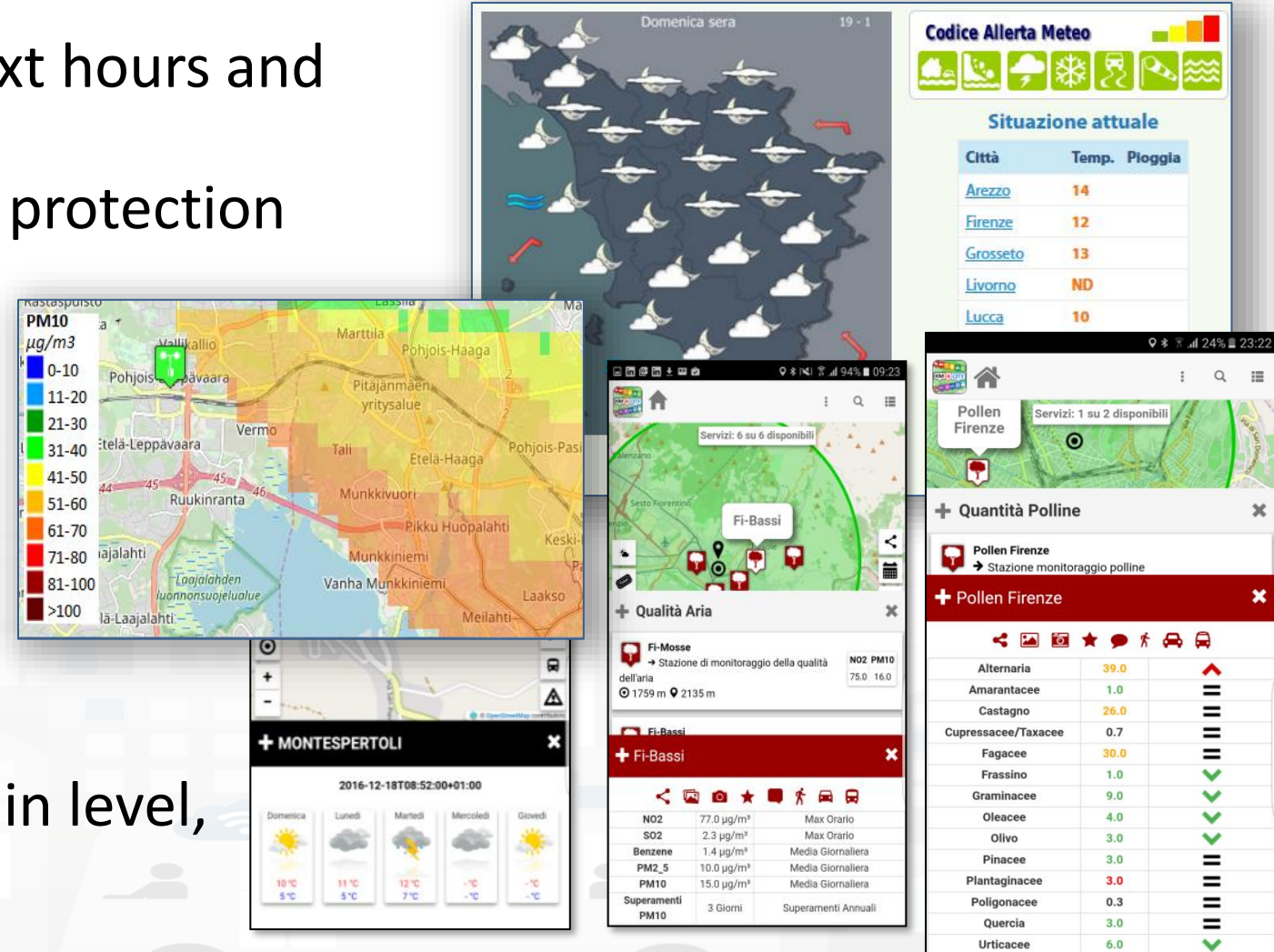
Stato Priorità	1°	2°	3°	4°	5°
Con Destinazione	0	1	0	0	0
In Attesa	0	1	1	0	0
In Visita	3	10	11	2	0
Oss. Temporanea	2	11	10	1	0
Totali	5	23	22	3	0

20-03-2017 00:37

I CODICI DEL PRONTO SOCCORSO	
CODICE ROSSO EMERGENZA accesso immediato	Casi con pericolo di vita Il trattamento di questi pazienti avviene immediatamente in via prioritaria
CODICE GIALLO URGENZA accesso rapido	Casi con lesioni gravi ed eventuale alterazione di una o più funzioni vitali L'assistenza viene assicurata nel minor tempo possibile
CODICE VERDE URGENZA differibile	Casi in condizioni critiche, ma non in pericolo di vita L'assistenza viene assicurata dopo i casi più urgenti
CODICE AZZURRO NON URGENZA	Casi in condizioni non gravi La prestazione sanitaria è differibile
CODICE BIANCO	Casi con problematiche risolvibili dal medico curante, dalla guardia medica o da ambulatori specialistici Tempi di attesa molto lunghi

Access at Environmental information

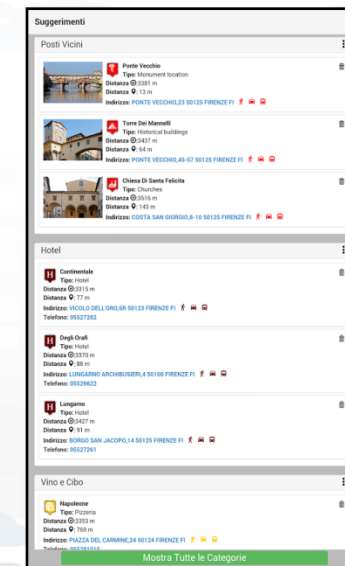
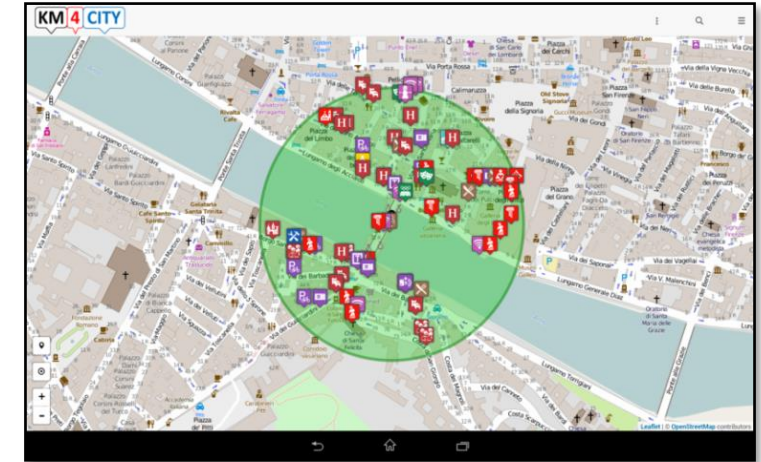
- Getting weather forecast for the next hours and days
- Getting alert information from Civil protection
- Getting air quality status
- Getting Air quality via heatmaps, heatmap animation
- Computing Air quality indexes
- Computing Air quality predictions
- Getting pollination status
- getting actual weather status: temperature, humidity, pressure, rain level, etc.



Profiled Suggestions to City Users

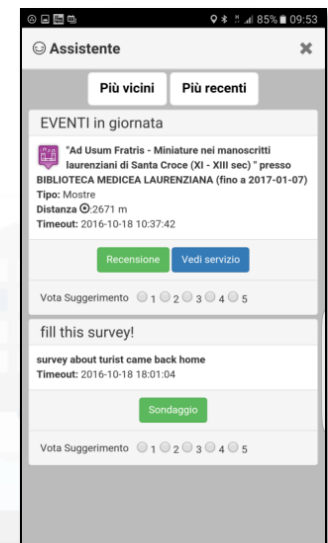
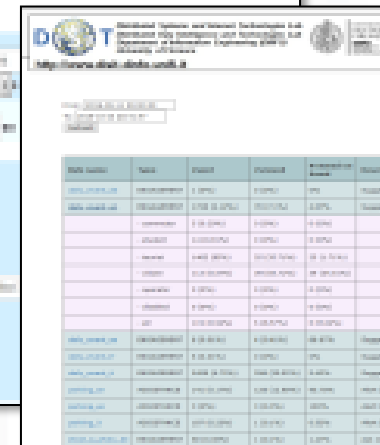
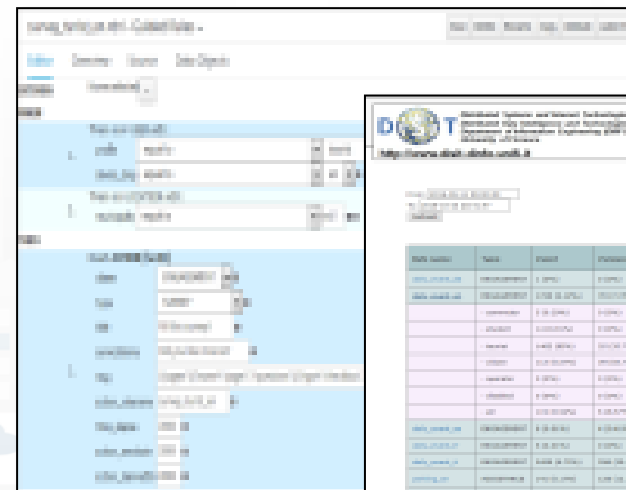
Personalized suggestions

- The server provide suggestions in the user context (location and time) arranged in a number of categories
 - Culture, mobility, food and drink, etc.
 - Alerts: civil protection, city council, twitter data, etc.
- The city user may reject some of them, thus the suggestion engine learns about preferred topics and category



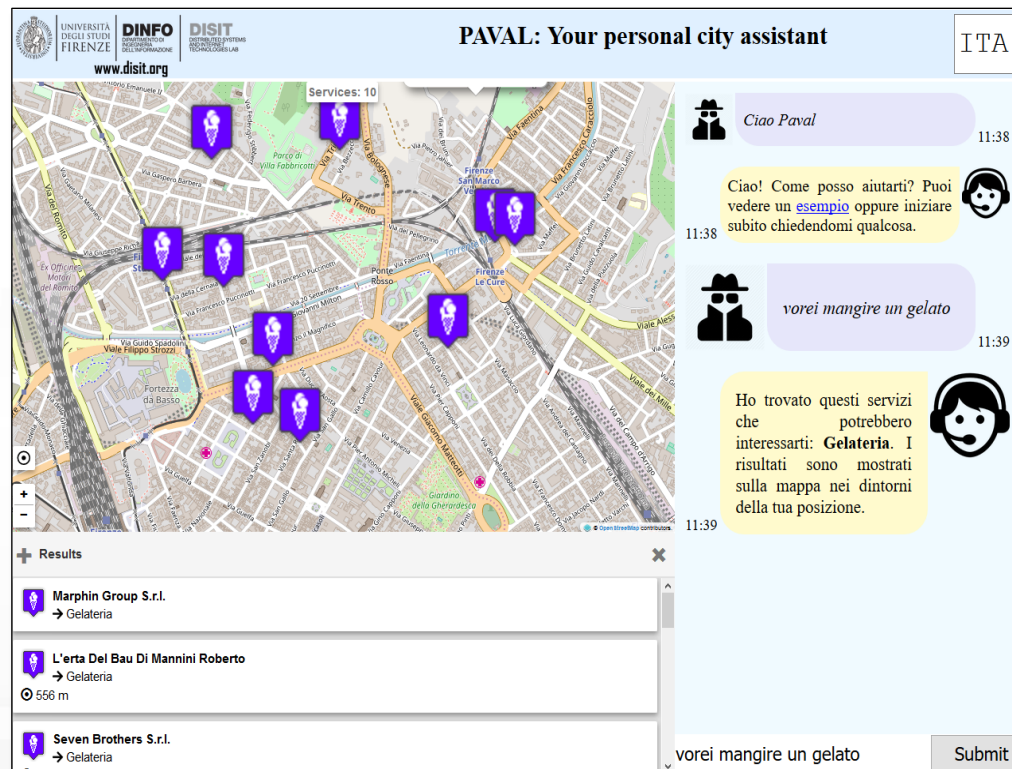
Profiled Engagements to City Users

- The user are profiled to learn habits:
 - Personal POI and paths
 - Mobility habits
- Information and engagements sent to the city users are programmed according to the user evolution to:
 - Stimulate virtuous habits
 - More sustainable habits
 - More healthy habits, etc.
 - Get feedbacks
 - Provide bonus and prices, ...
 - Send alerts, ...



PAVAL: Personal Assistant

- Your Personal Assistant for navigating in the city
- Ask PAVAl to get help and information about the city services
- ITA, ENG
- Active on Florence and whole Tuscany
- Mobile and PC



<https://assistant.disit.org>



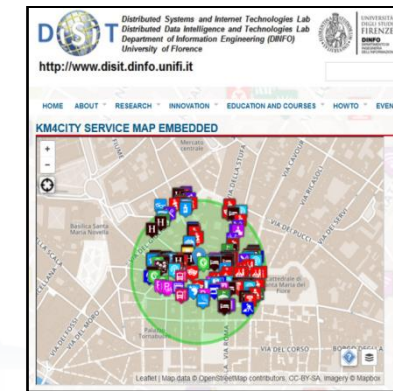
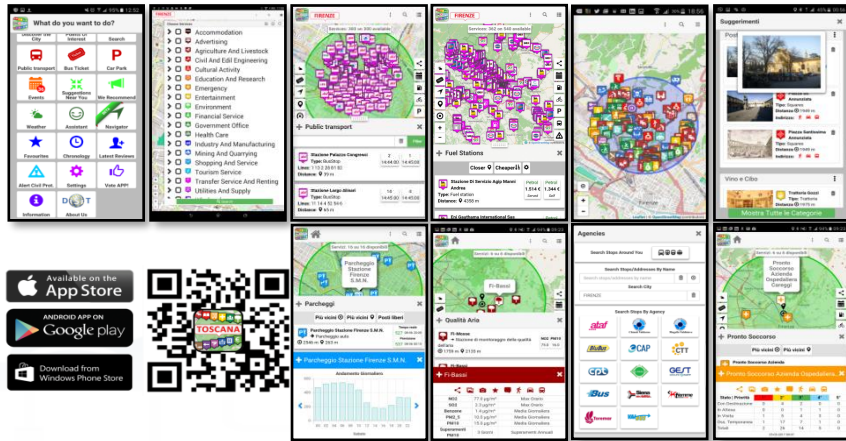


Developing Web and Mobile Apps, MicroApps,..

Mobile Apps

Web App HTML5, MicroApplications

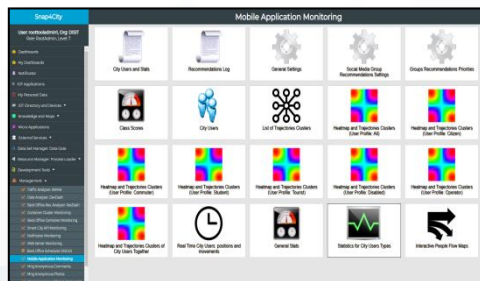
Embed into Web pages



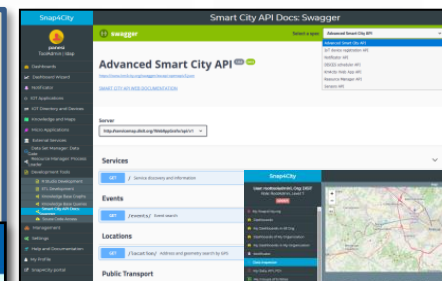
City User



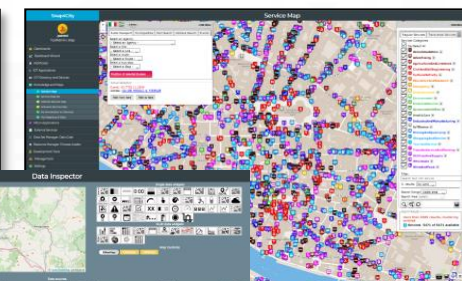
Advanced Smart City API



Snap/Km4City
Open Source
development
tool kit



Swagger



ServiceMap

DataInspector

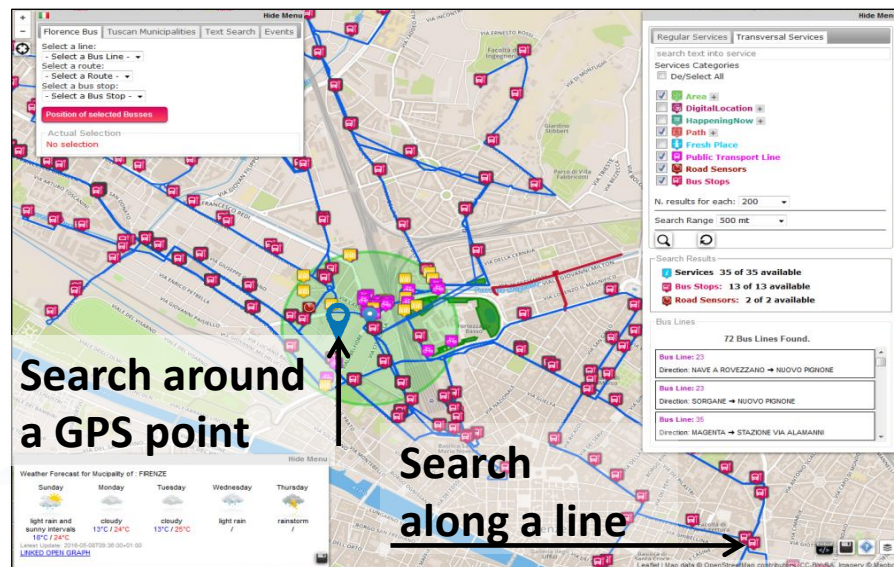
Developer



Mobile Application
Monitoring
Administrator



ServiceMap Dev Tool (knowledge & Map tool)

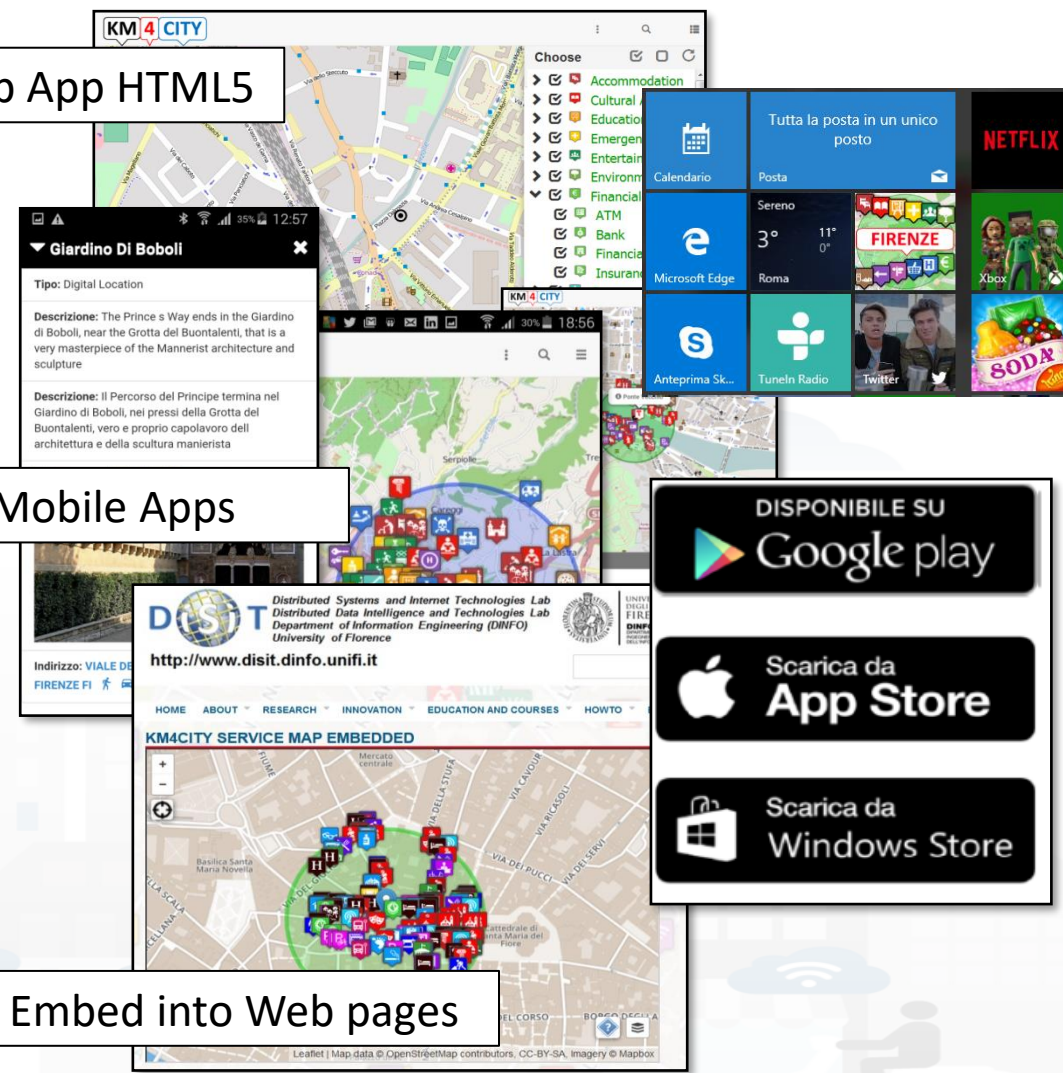


Smart City API call generation

Web App HTML5

Mobile Apps

Embed into Web pages



<http://www.disit.org/6873>

- based on Km4City engine on the back
- documented: <https://www.snap4city.org/404>
- **ServiceMap** tool to generate visually calls to exploit the Smart City API in web and mobile applications
- **Documentation and examples:**
 - [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](#)
- **The Alternatives:**
 - just Dashboards directly exploiting data on graphics and/or
 - IOT Applications via Node-RED exploiting MicroServices also using the Smart City APIs

TOP

Federated Knowledge Bases and Smart City APIs

FROM CITY
DASHBOARD
APPLICATION

DATA GATHERING
AND CITY DATA
KNOWLEDGE
MANAGEMENT

FORGING &
MANAGING OPEN
AND FLEXIBLE WEB
AND MOBILE APPS

IOT/IOE DEVICES
AND NETWORKS

IOT APPLICATIONS,
THE LOGIC AND
THE SMARTNESS

ADVANCED
SMART CITY API,
MICROSERVICES,
SNAP4CITY API

SNAP4CITY
LIVING LAB FOR
COLLABORATIVE
WORK

SNAP4CITY FOR
BEGINNERS

DATA ANALYTICS,
BUSINESS
INTELLIGENCE,
WHAT-IF AND
SIMULATIONS

SNAP4CITY
ARCHITECTURE AND
ECOSYSTEM OPENED
TO DEVELOPERS
AND STAKEHOLDERS

TWITTER
VIGILANCE: SOCIAL
MEDIA ANALYSIS

DECISION SUPPORT
SYSTEM AND CITY
RESILIENCE

HOW TO ADOPT
SNAP4CITY, AND
OUR ROADMAP

SNAP4CITY
AND KM4CITY
PROJECTS

SNAP4CITY THE
VIEW OF THE
ADMINISTRATORS

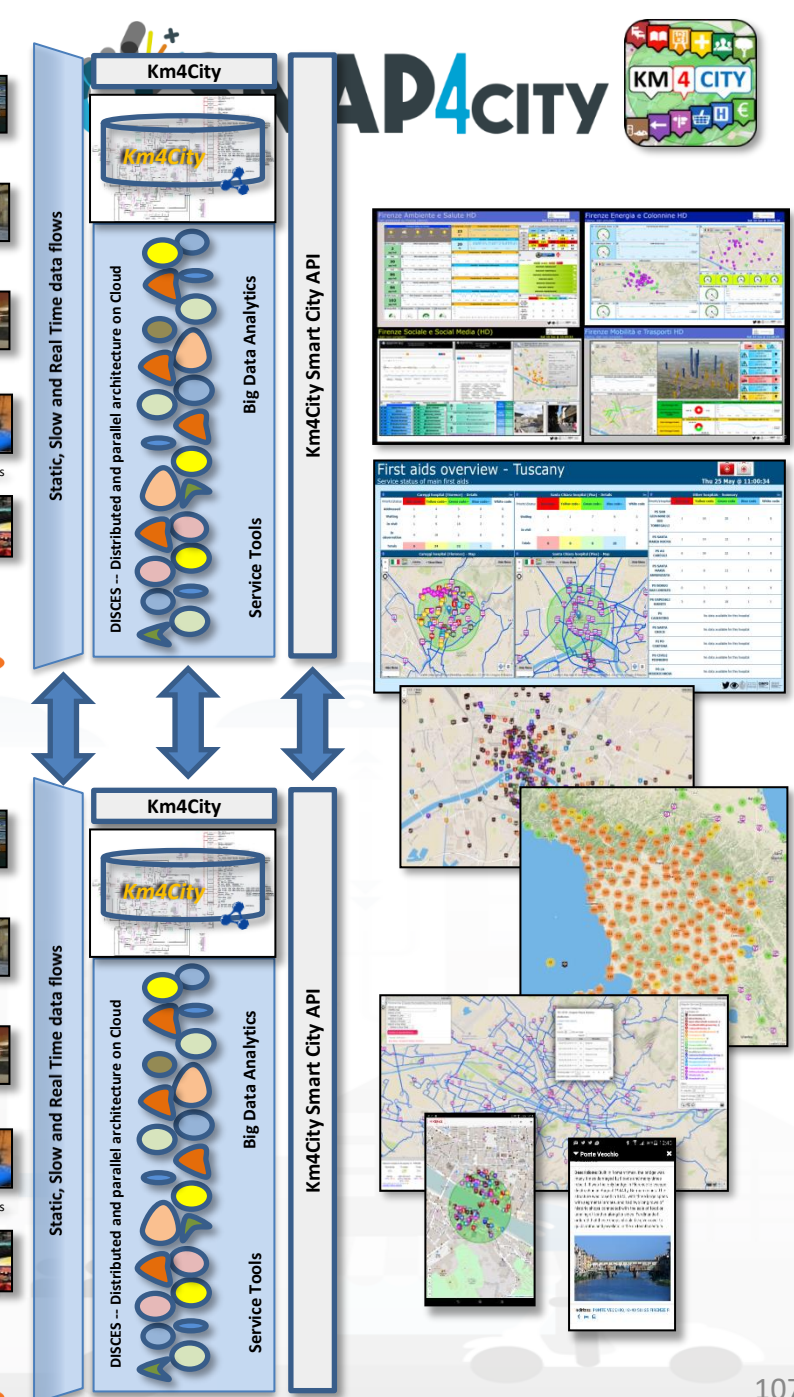
Km4City Federation

At different levels:

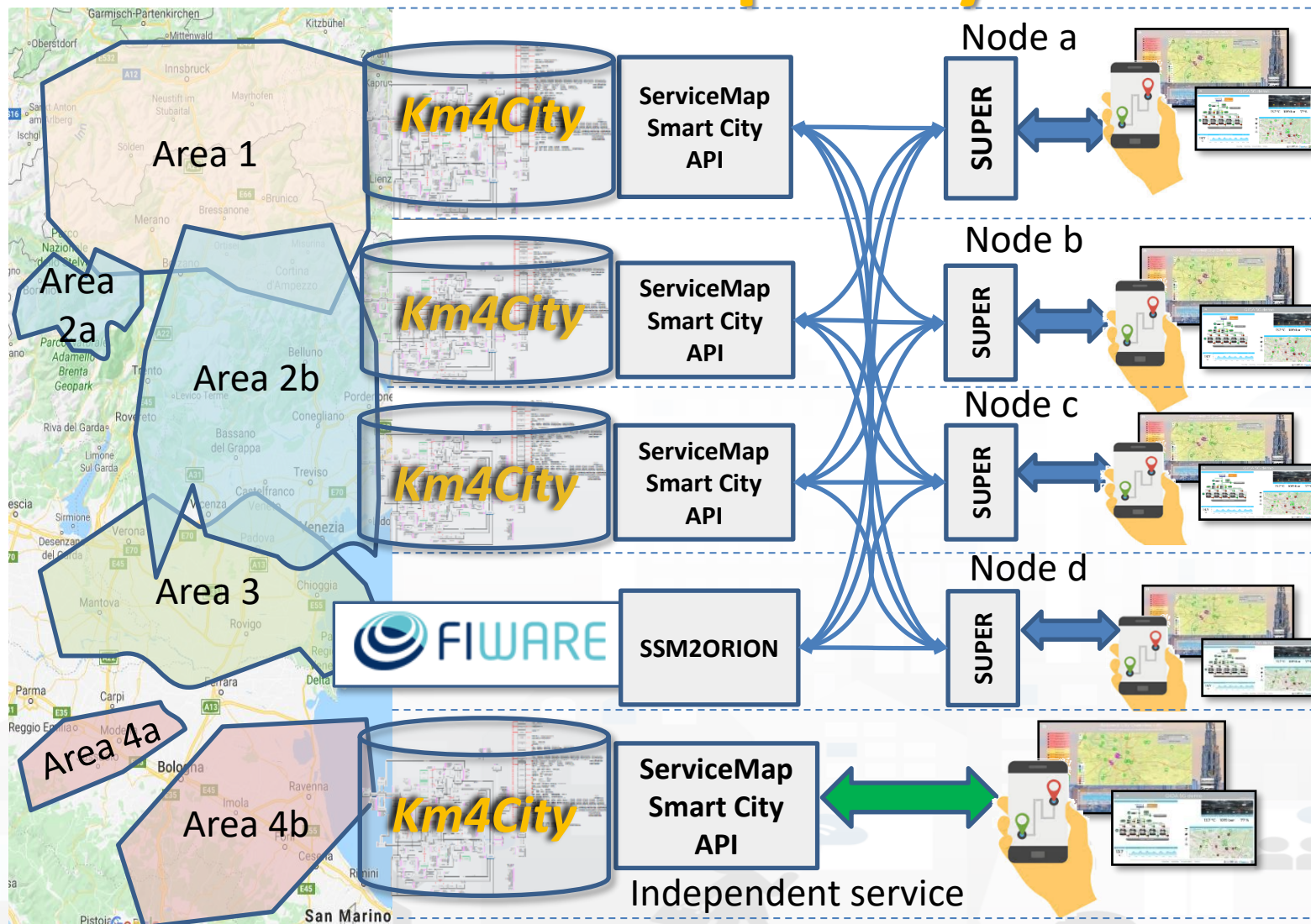
- Among cities/regions
- Among data providers
- Among Operators

By Means of:

- Smart City API → Apps
- Km4City Smart City Ontology
- Dashboards/data analytics

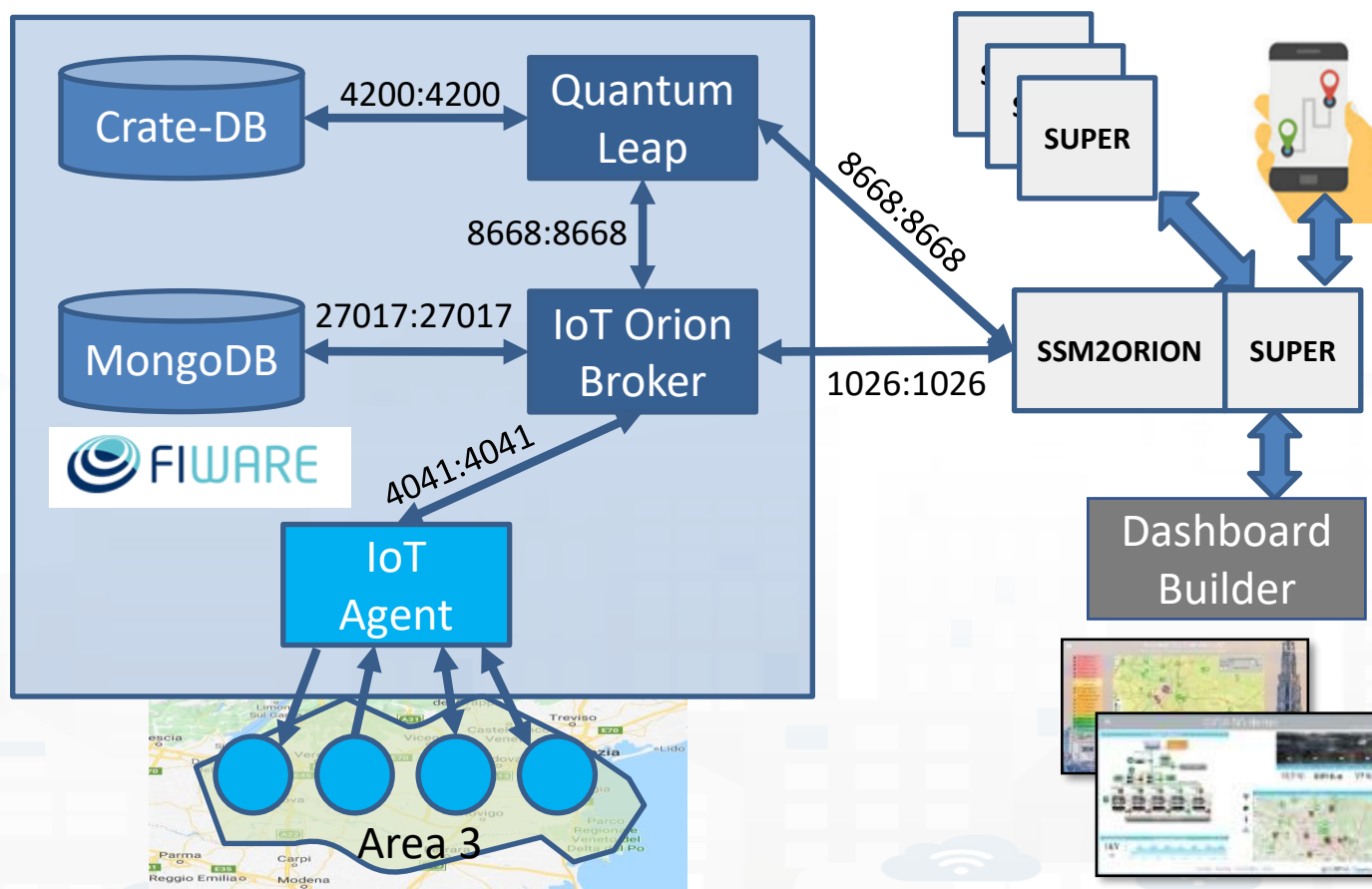


Federation of Snap4City Services



- 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 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 is not mandatory so that separate services can be produced as well
- Super and Nodes presents the same Smart City APIs.

Federation of Snap4City vs IOT ORION Broker



- 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 disjointed nodes, federation and independent services

Federated ServiceMap and Smart City API

To improve scalability, fault tolerance and federation among cities:

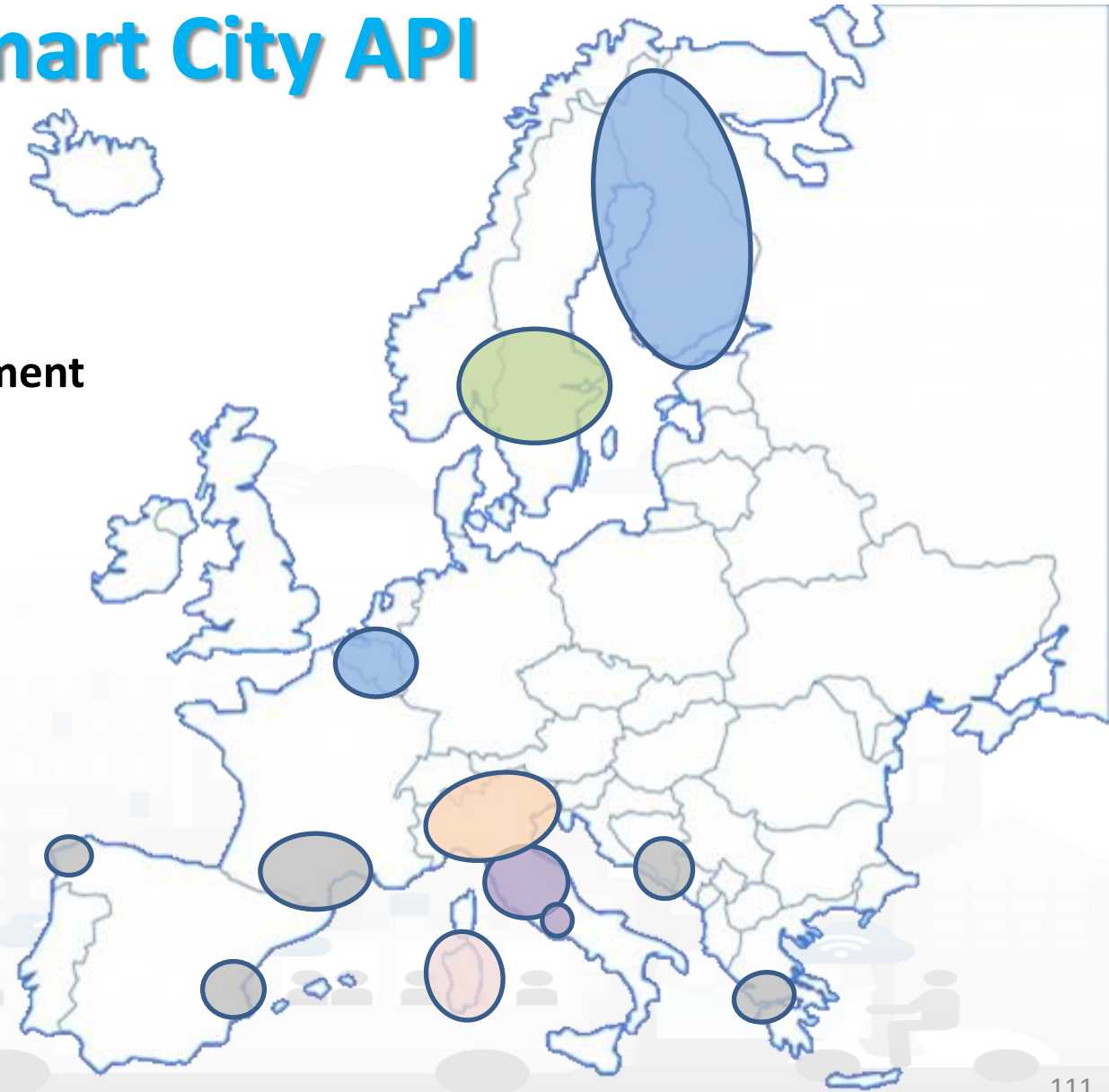
- One entry point Smart City API for all zones
- Multiple Knowledge base See performance assessment

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
- CKAN via harvesting



Exposing Services

- **Advanced Smart City API** which can be confined into a single Smart City installation or Federated as well as for Super Service Map
 - <https://www.km4city.org/swagger/external/index.html>
- **Federated Multiple Snap4City Knowledge Bases.** This allows the creation of mobile applications that may move from multiple cities and area accessing data and making queries transparently. This solution is presently in place among the Knowledge Bases of: Antwerp/Helsinki, Tuscany/Firenze, Sardegna, etc. The resulting Service is called Super Service Map and it is integrated in the Smart City API. For example, via:
 - <https://www.disit.org/superservicemap/api/v1>
- **Federated Open Data Portals** via DataGate/CKAN that presently presents now more than 13800 data sets linked for the cities of Helsinki and Antwerp.
 - <https://datagate.snap4city.org/organization>
 - Federation, Harvesting interface is: <https://datagate.snap4city.org/harvest>
- **WFS service of Snap4City** on top of Federated Smart City API or simple Smart City API of a single ServiceMap (smart City installation). This solution permits to GIS applications and platforms (such as ArcGIS OnLine ESRI, ArcGIS Enterprise ESRI, ArcGIS Map/pro Desktop, QGIS, GeoServer, etc.) to access at Snap4City data. For Example, via:
 - <https://www.disit.org/superservicemap/api/v1/wfs>
 - <https://www.disit.org/superservicemap/api/v1/wfs?service=WFS&request=GetCapabilities&version=2.0.0>
- **WMS service of Snap4City** for publishing **maps and heatmaps**, provided by an installed GeoServer third party open source tool. For example, via:
 - <https://wmserver.snap4city.org/geoserver/Snap4City/wms>
 - <https://www.km4city.org/swagger/external/index.html?urls.primaryName=Heatmap%20API>

TOP

Web and Mobile App Development Kit

FROM CITY DASHBOARD TO APPLICATIONS

DATA GATHERING AND CITY DATA KNOWLEDGE MANAGEMENT

FORGING & MANAGING OPEN AND FLEXIBLE WEB AND MOBILE APPS

IOT/IOE DEVICES AND NETWORKS

APPLICATIONS THE LOGIC OF THE SMARTNESS

ADVANCE SMART CITY API, SMART SERVICES, SNAP4CITY API

SNAP4CITY LIVING LAB FOR COLLABORATIVE WORK

DATA ANALYTICS, BUSINESS INTELLIGENCE, WHAT-IF AND SIMULATION

SNAP4CITY FOR BUSINESS

SNAP4CITY ARCHITECTURE AND ECOSYSTEM OPENED TO DEVELOPERS AND PARTNERS

TWITTER VIGILANCE: SOCIAL MEDIA ANALYSIS

HOW TO ADOPT SNAP4CITY, AND OUR ROADMAP

SNAP4CITY AND KM4CITY PROJECTS

SNAP4CITY THE VIEW OF THE ADMINISTRATORS



APACHE
CORDOVA™

- **Apache Cordova** is a set of **JavaScript APIs** that enable the developer to access native features of the device such as the camera or accelerometer, storage, network, gps
- Combined with a user **interface framework** such as Dojo Mobile or jQuery Mobile or Sencha Touch, allows the development of smartphone applications using only **HTML, CSS and JavaScript**.
- When using the Cordova API, an application can be built without any native code (Java, Objective-C, C# etc.). The **web technologies** used are **hosted in the same application** at the local level (usually not on a remote http server).
- These **JavaScript API** are **consistent** and **valid** for the **different platforms** of mobile devices, in this way the application built on the Web standard, should be **portable** with a **minimum of changes**.

Mustache JS

- The library is **independent** from specific framework but there are plugins for the integration with jQuery, Dojo, and YUI.
- Possibility to work with **javascript objects** and then exploit the communication of **data** in **JSON format** from a **REST** call via AJAX.
- The **templates** for Mustache may be assigned or loaded as a string to a variable and the placeholder are identified by two braces, for example: {{miopplaceholder}}.
- One of the most interesting of the library feature is support in **enumerable values**
- Documentation and downloads are available on the official website:
<http://mustache.github.io>

Mustache JS

Template



```
<h1>{{titolo}}</h1>
<p>{{descrizione}}</p>
{{#risultato}} //solo se risultato è true
  <ul>{{#citta}}
    <li> {{nome}} ({{sigla}})</li>
  </ul>
</risultato>
{{^risultato}} //altrimenti...
  <p><em>Nessuna città trovata!</em></p>
</risultato>
```

JSON



```
var data = {
  risultato: true,
  titolo: Città italiane,
  descrizione: Lista delle città italiane,
  citta: [
    {nome: Milano, sigla: MI},
    {nome: Roma, sigla: RM}
  ]
};
```


Mustache JS

Template



```
<h1>{{titolo}}</h1>
<p>{{descrizione}}</p>
{{#risultato}} //solo se risultato è true
  <ul>{{#citta}}
    <li> {{nome}} ({{sigla}})</li>
  </ul>
</risultato>
{{^risultato}} //altrimenti...
  <p><em>Nessuna città trovata!</em></p>
</risultato>
```

JSON

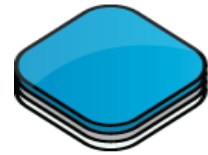


```
var data = {
  risultato: true,
  titolo: Città italiana,
  descrizione: Lista delle città italiane,
  citta: [
    {nome: Milano, sigla: MI},
    {nome: Roma, sigla: RM}
  ]
};
```

Template + JSON + Mustache

Città italiana
Lista delle città italiane

- Milano (MI)
- Roma (RM)

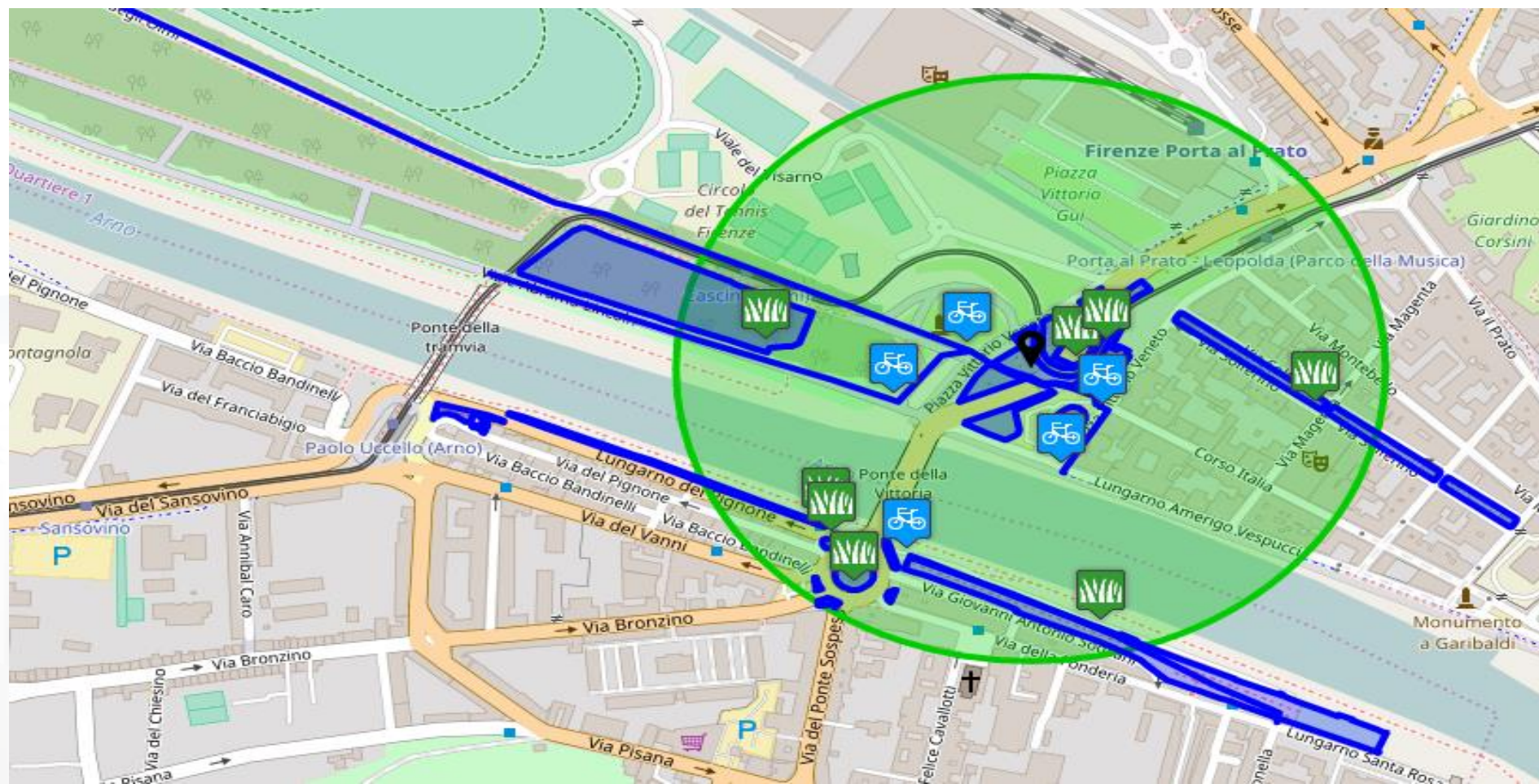


OpenLayers 3.0

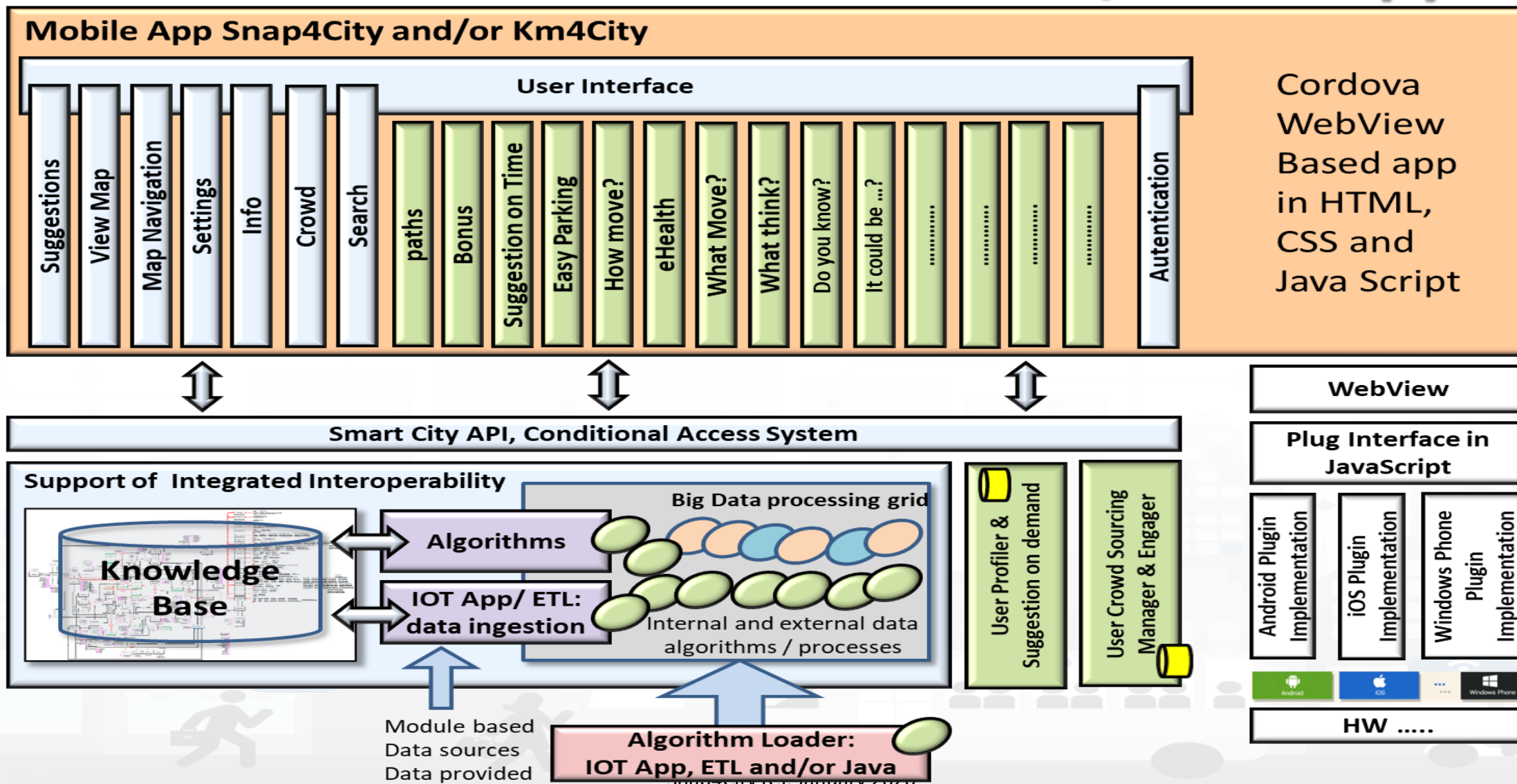
- **OpenLayers** is an open source **JavaScript library** for **displaying map data** in web browsers and can be used with a hybrid application developed with Cordova
- In the **early versions** of the app, the map was managed by **Leaflet.js** library. This was replaced because it didn't support the rotation, which is required to insert navigation functions within the app
- In addition, OpenLayers 3.0 builds the map and objects added to it with a **canvas** renderer, which is **very efficient** when objects are **numerous and small** as the markers displayed for each search done with the app
- Documentation and downloads are available on the official website: **<http://openlayers.org>**



OpenLayers 3.0



General architecture of Mobile / Web App



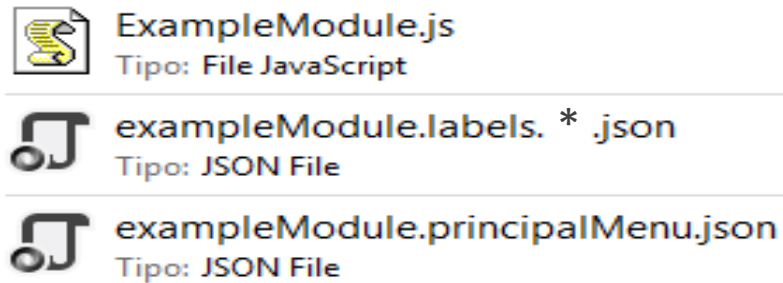
Create ParkingSearcher Module

In the slides following there is an **example** of how to add a **module** to the app.

The goal of this example is to create a **new module** that in addition to viewing the list of car parks as is already the case for the button named "Parking" will **show directly** the **number of free parking lots** for each car park found

Create ParkingSearcher Module

- Files required for creating a new module are as follows



A **Javascript** file containing the **logic**








5 JSON files (**ita, eng, esp, deu, fra**) containing **labels** to be included in the new interface

A JSON file that contains one or more **buttons** to be added to **principal menu** to allow the user to interact with the newly created module

Create ParkingSearcher Module

- Copy these files to a **new folder** that will have the **name of the new module** (i.e., **ParkingSearcher**): the **names of the files** copied have to be changed to get the **module name** as a **prefix**

oro > workspace > siiMobilityAppKit > www > js > modules > parkingSearcher

	ParkingSearcher.js Tipo: File JavaScript
	parkingSearcher.labels.deu.json Tipo: JSON File
	parkingSearcher.labels.eng.json Tipo: JSON File
	parkingSearcher.labels.fra.json Tipo: JSON File
	parkingSearcher.labels.ita.json Tipo: JSON File
	parkingSearcher.labels.spa.json Tipo: JSON File
	parkingSearcher.principalMenu.json Tipo: JSON File

ParkingSearcher in main menu

- Field descriptions for creating buttons in the main menu

```
[
{
  "callback": "PrincipalMenu.hide(): MapManager.centerMapOnGps():".
  "iconId": "",
  "iconClass": "icon ion-android-bus",
  "iconFontSize": "41px",
  "iconColor": "#CC0000",
  "imgSrc": "img/ticketmenu.png",
  "imgHeight": "37px",
  "text": "P",
  "textFontSize": "38px",
  "textColor": "#CC0000",
  "captionId": "principalMenuParkingSearcher",
  "captionTextId": "moduleParkingSearcher",
  "step": true,
  "stepId": "eventsBadge",
  "ribbon": true,
  "ribbonId": "",
  "ribbonStyle": "background: #336633;background: linear-gradient(#33FF33 0%, #336633 100%);",
  "ribbonText": "Beta",
  "removed": false,
  "index": 0
}
]
```

parkingSearcher.principalMenu.json

This field contains the **callback** for the new module.

The present callbacks should be left, because they serves to **close the main menu** and to **center the map on the GPS**

ParkingSearcher in main menu

- Field descriptions for creating buttons in the main menu

```
[
{
  "callback": "PrincipalMenu.hide(); MapManager.centerMapOnGps();",
  "iconId": "",
  "iconClass": "icon ion-android-bus",
  "iconFontSize": "41px",
  "iconColor": "#CC0000",
  "imgSrc": "img/ticketmenu.png",
  "imgHeight": "37px",
  "text": "P",
  "textFontSize": "38px",
  "textColor": "#CC0000",
  "captionId": "principalMenuParkingSearcher",
  "captionTextId": "moduleParkingSearcher",
  "step": true,
  "stepId": "eventsBadge",
  "ribbon": true,
  "ribbonId": "",
  "ribbonStyle": "background: #336633;background: linear-gradient(#33FF33 0%, #336633 100%);",
  "ribbonText": "Beta",
  "removed": false,
  "index": 0
}
]
```

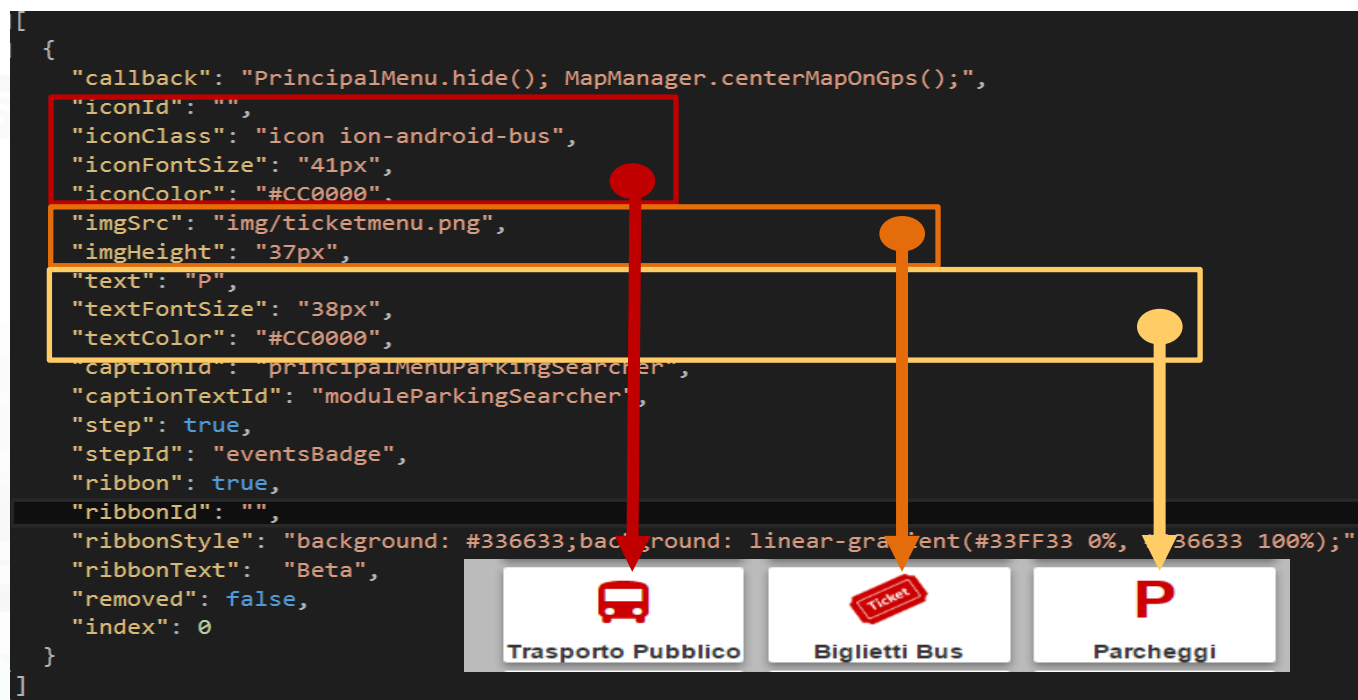
parkingSearcher.principalMenu.json

These blocks of fields are **mutually exclusive**. Allow you to choose the icon that will identify the button that you are creating. This icon can be chosen as an **image**, a **text**, a **glyphicon** (Bootstrap) or **ionicons** (ionicons.com).

N.B. Field **iconId** can be useful if you plan to edit the selected icon **dynamically**

ParkingSearcher in main menu

- Field descriptions for creating buttons in the main menu



parkingSearcher.principalMenu.json

These blocks of fields are **mutually exclusive**. Allow you to choose the icon that will identify the button that you are creating. This icon can be chosen as an **image**, a **text**, a **glyphicon** (Bootstrap) or **ionicons** (ionicons.com).

N.B. Field **iconId** can be useful if you plan to edit the selected icon **dynamically**

ParkingSearcher in main menu

- Field descriptions for creating buttons in the main menu

```
[
{
  "callback": "PrincipalMenu.hide(); MapManager.centerMapOnGps();",
  "iconId": "",
  "iconClass": "icon ion-android-bus",
  "iconFontSize": "41px",
  "iconColor": "#CC0000",
  "imgSrc": "img/ticketmenu.png",
  "imgHeight": "37px",
  "text": "P",
  "textFontSize": "38px",
  "textColor": "#CC0000",
  "captionId": "principalMenuParkingSearcher",
  "captionTextId": "moduleParkingSearcher",
  "step": true,
  "stepId": "eventsBadge",
  "ribbon": true,
  "ribbonId": "",
  "ribbonStyle": "background: #336633;background: linear-gradient(#33FF33 0%, #336633 100%);",
  "ribbonText": "Beta",
  "removed": false,
  "index": 0
}
]
```

parkingSearcher.principalMenu.json

captionId serves to indicate the **container tag** of the text that is located at the bottom of each button.

captionTextId indicates the name of the field in labels.*.json whose value is the text to be inserted in the previous container.

ParkingSearcher in main menu

- Field descriptions for creating buttons in the main menu

```
[
{
  "callback": "PrincipalMenu.hide(); MapManager.centerMapOnGps();",
  "iconId": "",
  "iconClass": "icon ion-android-bus",
  "iconFontSize": "41px",
  "iconColor": "#CC0000",
  "imgSrc": "img/ticketmenu.png",
  "imgHeight": "37px",
  "text": "P",
  "textFontSize": "38px",
  "textColor": "#CC0000",
  "captionId": "principalMenuParkingSearcher",
  "captionTextId": "moduleParkingSearcher",
  "step": true,
  "stepId": "eventsBadge",
  "ribbon": true,
  "ribbonId": "",
  "ribbonStyle": "background: #336633;background: linear-gradient(#33FF33 0%, #336633 100%);",
  "ribbonText": "Beta",
  "removed": false,
  "index": 0
}
]
```

parkingSearcher.principalMenu.json

These blocks of fields are used to show the user **badges containing information** related to the button on which are located

ParkingSearcher in main menu

- Field descriptions for creating buttons in the main menu

```
[
{
  "callback": "PrincipalMenu.hide(); MapManager.centerMapOnGps();",
  "iconId": "",
  "iconClass": "icon ion-android-bus",
  "iconFontSize": "41px",
  "iconColor": "#CC0000",
  "imgSrc": "img/ticketmenu.png",
  "imgHeight": "37px",
  "text": "P",
  "textFontSize": "38px",
  "textColor": "#CC0000",
  "captionId": "principalMenuParkingSearcher",
  "captionTextId": "moduleParkingSearcher",
  "step": true,
  "stepId": "eventsBadge",
  "ribbon": true,
  "ribbonId": "",
  "ribbonStyle": "background: #336633;background: linear-gradient(#33FF33 0%, #336633 100%);",
  "ribbonText": "Beta",
  "removed": false,
  "index": 0
}
]
```



These blocks of fields are used to show the user **badges containing information** related to the button on which are located

parkingSearcher.principalMenu.json

ParkingSearcher in main menu

- Field descriptions for creating buttons in the main menu

```
[
{
  "callback": "PrincipalMenu.hide(); MapManager.centerMapOnGps();",
  "iconId": "",
  "iconClass": "icon ion-android-bus",
  "iconFontSize": "41px",
  "iconColor": "#CC0000",
  "imgSrc": "img/ticketmenu.png",
  "imgHeight": "37px",
  "text": "P",
  "textFontSize": "38px",
  "textColor": "#CC0000",
  "captionId": "principalMenuParkingSearcher",
  "captionTextId": "moduleParkingSearcher",
  "step": true,
  "stepId": "eventsBadge",
  "ribbon": true,
  "ribbonId": "",
  "ribbonStyle": "background: #336633;background: linear-gradient(#33FF33 0%, #336633 100%);",
  "ribbonText": "Beta",
  "removed": false,
  "index": 0
}
]
```

parkingSearcher.principalMenu.json

removed field is useful to allow the removal and the insertion of the buttons in the main menu by the user.

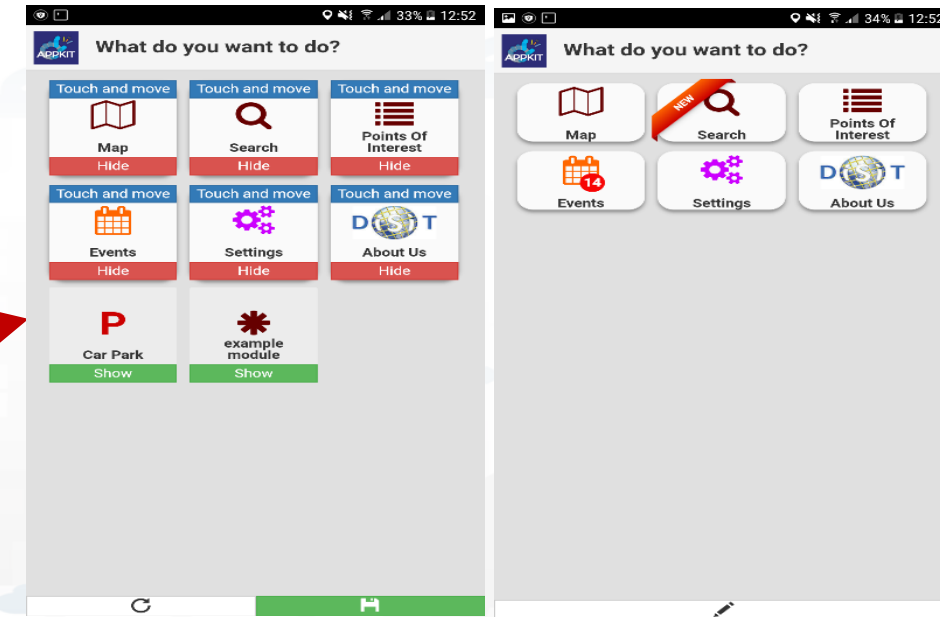
index field is useful for rendering the buttons in the order chosen by the user.

ParkingSearcher in main menu

- Field descriptions for creating buttons in the main menu

```
[
{
  "callback": "PrincipalMenu.hide(); MapManager.centerMapOnGps();",
  "iconId": "",
  "iconClass": "icon ion-android-bus",
  "iconFontSize": "41px",
  "iconColor": "#CC0000",
  "imgSrc": "img/ticketmenu.png",
  "imgHeight": "37px",
  "text": "P",
  "textFontSize": "38px",
  "textColor": "#CC0000",
  "captionId": "principalMenuParkingSearcher",
  "captionTextId": "moduleParkingSearcher",
  "step": true,
  "stepId": "eventsBadge",
  "ribbon": true,
  "ribbonId": "",
  "ribbonStyle": "background: #336633; background: linear-gradient(#33FF33 0%, #336633 100%);",
  "ribbonText": "Beta",
  "removed": false,
  "index": 0
}
]
```

parkingSearcher.principalMenu.json



ParkingSearcher in main menu

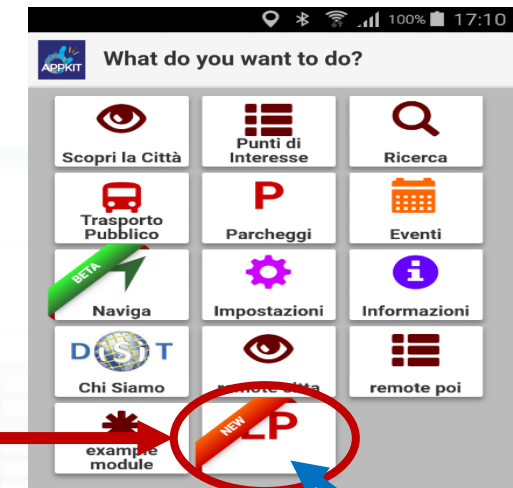
- Loading **new buttons modules** within the main menu, takes place by **comparing the captionId** field.
- If the menu already has a button with the **same captionId**, the first is **replaced** with the **new one**.
- To **remove** a button from the main menu (field **removed** hides it) add a **delete** field with value equal to **true**.

ParkingSearcher in main menu

- First version of the button

```
[
{
  "callback": "PrincipalMenu.hide(); MapManager.centerMapOnGps();",
  "iconId": "",
  "iconClass": "",
  "iconFontSize": "",
  "iconColor": "",
  "imgSrc": "",
  "imgHeight": "",
  "text": "LP",
  "textFontSize": "38px",
  "textColor": "#CC0000",
  "captionId": "principalMenuParkingSearcher",
  "captionTextId": "moduleParkingSearcher",
  "step": "",
  "stepId": "",
  "ribbon": true,
  "ribbonId": "",
  "ribbonStyle": "background: #CC0000;background: linear-gradient(#FF6600 0%, #CC0000 100%);",
  "ribbonText": "NEW",
  "removed": false,
  "index": 0
}
]
```

parkingSearcher.principalMenu.json



Label missing

Labels of ParkingSearcher

- Description of **label.*.json** files

label.ita.json

```
{  
  "principalMenu": {  
    "moduleParkingSearcher": "Lista Parcheggi"  
  }  
}
```

label.eng.json

```
{  
  "principalMenu": {  
    "moduleParkingSearcher": "Car Park List"  
  }  
}
```

label.deu.json

```
{  
  "principalMenu": {  
    "moduleParkingSearcher": "Parkplatz Liste"  
  }  
}
```

label.fra.json

```
{  
  "principalMenu": {  
    "moduleParkingSearcher": "Liste parkings"  
  }  
}
```

label.esp.json

```
{  
  "principalMenu": {  
    "moduleParkingSearcher": "Lista de Aparcamiento"  
  }  
}
```

Three important things to check:

- Languages shall be indicated by 3 characters: **ita**, **deu**, **esp**, **fra**, **eng**
- The label for the button must be contained within the object "**principalMenu**"
- The name of the field inside "principalMenu" must be the same of "**captionTextId**" seen before

Labels of ParkingSearcher

- Description of **label.*.json** files

```
[
{
  "callback": "PrincipalMenu.hide(); MapManager.centerMapOnGps();",
  "iconId": "",
  "iconClass": "",
  "iconFontSize": "",
  "iconColor": "",
  "imgSrc": "",
  "imgHeight": "",
  "text": "LP",
  "textFontSize": "38px",
  "textColor": "#CC0000",
  "captionId": "principalMenuParkingSearcher",
  "captionTextId": "moduleParkingSearcher",
  "step": "",
  "stepId": "",
  "ribbon": true,
  "ribbonId": "",
  "ribbonStyle": "background: #CC0000;background: linear-gradient(#",
  "ribbonText": "NEW",
  "removed": false,
  "index": 0
}
]
```

parkingSearcher.principalMenu.json

label.ita.json

```
{
  "principalMenu": {
    "moduleParkingSearcher": "Lista Parcheggi"
  }
}
```

label.eng.json

```
{
  "principalMenu": {
    "moduleParkingSearcher": "Car Park List"
  }
}
```

label.deu.json

```
{
  "principalMenu": {
    "moduleParkingSearcher": "Parkplatz Liste"
  }
}
```

label.fra.json

```
{
  "principalMenu": {
    "moduleParkingSearcher": "Liste parkings"
  }
}
```

label.esp.json

```
{
  "principalMenu": {
    "moduleParkingSearcher": "Lista de Aparcamiento"
  }
}
```

```
$(captionId).html(
  labels.principalMenu[
    captionTextId]);
```


Labels of ParkingSearcher

- Description of **label.*.json** files

```
[
  {
    "callback": "PrincipalMenu.hide(); MapManager.centerMapOnGps();",
    "iconId": "",
    "iconClass": "",
    "iconFontSize": "",
    "iconColor": "",
    "imgSrc": "",
    "imgHeight": "",
    "text": "LP",
    "textFontSize": "38px",
    "textColor": "#CC0000",
    "captionId": "principalMenuParkingSearcher",
    "captionTextId": "moduleParkingSearcher",
    "step": "",
    "stepId": "",
    "ribbon": true,
    "ribbonId": "",
    "ribbonStyle": "background: #CC0000;background: linear-gradient(#CC0000, #CC0000);",
    "ribbonText": "NEW",
    "removed": false,
    "index": 0
  }
]
```

parkingSearcher.principalMenu.json

label.ita.json

```
{
  "principalMenu": {
    "moduleParkingSearcher": "Lista Parcheggi"
  }
}
```



label.eng.json

```
{
  "principalMenu": {
    "moduleParkingSearcher": "Car Park List"
  }
}
```



label.deu.json

```
{
  "principalMenu": {
    "moduleParkingSearcher": "Parkplatz Liste"
  }
}
```



label.fra.json

```
{
  "principalMenu": {
    "moduleParkingSearcher": "Liste parkings"
  }
}
```



label.esp.json








```
{
  "principalMenu": {
    "moduleParkingSearcher": "Lista de Aparcamiento"
  }
}
```



Create ParkingSearcher Module

- It is seen as fill most of the files in the folder of new module ParkingSearcher that is developed in this presentation

oro > workspace > siiMobilityAppKit > www > js > modules > parkingSearcher

	ParkingSearcher.js Tipo: File JavaScript	TODO
	parkingSearcher.labels.deu.json Tipo: JSON File	✓
	parkingSearcher.labels.eng.json Tipo: JSON File	✓
	parkingSearcher.labels.fra.json Tipo: JSON File	✓
	parkingSearcher.labels.ita.json Tipo: JSON File	✓
	parkingSearcher.labels.spa.json Tipo: JSON File	✓
	parkingSearcher.principalMenu.json Tipo: JSON File	✓

ParkingSearcher Module Functions

- Functions contained in **ParkingSearcher.js**

```
show: function () {  
    application.resetInterface();  
    MapManager.showMenuReduceMap("#" + ParkingSearcher.idMenu);  
    $("#" + ParkingSearcher.idMenu + "Collapse").hide();  
    ParkingSearcher.open = true;  
    InfoManager.addingMenuToManage(ParkingSearcher.varName);  
    application.addingMenuToCheck(ParkingSearcher.varName);  
    application.setBackButtonListener();  
},
```

```
hide: function () {  
    $("#" + ParkingSearcher.idMenu).css({ 'z-index': '1001' });  
    MapManager.reduceMenuShowMap("#" + ParkingSearcher.idMenu);  
    InfoManager.removingMenuToManage(ParkingSearcher.varName);  
    application.removingMenuToCheck(ParkingSearcher.varName);  
    ParkingSearcher.open = false;  
},
```

Closes any previously **opened menu**, **shrinks the map** to display the menu, **hides the button** to reduce the menu, since it will open already reduced.

Recording to other variables to get notifications when:

- users press the **back button**
- users change the **device orientation**
- must be **closed the menu** opened by this module

ParkingSearcher Module Functions

- Functions contained in **ParkingSearcher.js**

```
show: function () {  
    application.resetInterface();  
    MapManager.showMenuReduceMap("#" + ParkingSearcher.idMenu);  
    $("#" + ParkingSearcher.idMenu + "Collapse").hide();  
    ParkingSearcher.open = true;  
    InfoManager.addingMenuToManage(ParkingSearcher.varName);  
    application.addingMenuToCheck(ParkingSearcher.varName);  
    application.setBackButtonListener();  
},
```

```
hide: function () {  
    $("#" + ParkingSearcher.idMenu).css({ 'z-index': '1001' });  
    MapManager.reduceMenuShowMap("#" + ParkingSearcher.idMenu);  
    InfoManager.removingMenuToManage(ParkingSearcher.varName);  
    application.removingMenuToCheck(ParkingSearcher.varName);  
    ParkingSearcher.open = false;  
},
```

Does the **opposite functions** to those performed by the **function show**, also reset the z-index of the menu

ParkingSearcher Module Functions

- Functions contained in **ParkingSearcher.js**

```
checkForBackButton: function () {  
    if (ParkingSearcher.open) {  
        ParkingSearcher.hide();  
    }  
},  
  
refreshMenuPosition: function () {  
    if (ParkingSearcher.open) {  
        MapManager.showMenuReduceMap("#" + ParkingSearcher.idMenu);  
        Utility.checkAxisToDrag("#" + ParkingSearcher.idMenu);  
        if (ParkingSearcher.expanded) {  
            ParkingSearcher.expandBusRoutesMenu();  
        }  
    }  
},  
  
closeAll: function () {  
    if (ParkingSearcher.open) {  
        ParkingSearcher.hide();  
    }  
},
```

These are the **callbacks** called to **notify** the occurrence of an event among those described previously (see show function) and for which we recorded the module

- users press the **back button**
- users change the **device orientation**
- must be **closed the menu** opened by this module

ParkingSearcher Module Functions

- Functions contained in **ParkingSearcher.js**

```
refreshMenu: function () {  
    if ($("#" + ParkingSearcher.idMenu).length == 0) {  
        $("#indexPage").  
            append("<div id=\"" + ParkingSearcher.idMenu + "\" class=\"commonHalfMenu\"></div>")  
    }  
    ViewManager.render(ParkingSearcher.results, "#" + ParkingSearcher.idMenu, "ParkingMenu");  
    Utility.movingPanelWithTouch("#" + ParkingSearcher.idMenu + "ExpandHandler",  
        "#" + ParkingSearcher.idMenu);  
},
```

- Checks if there is the **element** that will **contain the html code** created through the use of **Mustache** library.
- It is generated the html code with **template ParkingMenu.mst.html** and **JSON ParkingSearcher.results** and added to the element container.
- Finally, the **feature** that allows the users **to widen the menu by dragging** the handler is added to it

ParkingSearcher Module Functions

- Functions contained in **ParkingSearcher.js**

```
refreshMenu: function () {  
    if ($("#" + ParkingSearcher.idMenu).length == 0) {  
        $("#indexPath").  
            append("<div id=\"" + ParkingSearcher.idMenu + "\" class=\"commonHalfMenu\"></div>")  
    }  
    ViewManager.render(ParkingSearcher.results, "#" + ParkingSearcher.idMenu, "ParkingMenu");  
    Utility.movingPanelWithTouch("#" + ParkingSearcher.idMenu + "ExpandHandler",  
        "#" + ParkingSearcher.idMenu);  
},
```

- Checks if there is the **element** that will **contain the html code** created through the use of **Mustache** library.
- It is generated the html code with **template ParkingMenu.mst.html** and **JSON ParkingSearcher.results** and added to the element container.
- Finally, the **feature** that allows the users **to widen the menu by dragging** the handler is added to it

ParkingSearcher Module Functions

- Functions contained in **ParkingSearcher.js**

```
successQuery: function (response) {  
    ParkingSearcher.results = responseObject["Results"];  
    ParkingSearcher.refreshMenu();  
    ParkingSearcher.show();  
    MapManager.addGeoJSONLayer(responseObject);  
    ParkingSearcher.resetSearch();  
},  
  
errorQuery: function(error) {  
    navigator.notification.alert(  
        Globalization.alerts.servicesServerError.message,  
        function () { },  
        Globalization.alerts.servicesServerError.title);  
},
```

These are the callbacks that should be called once the **JSON**, containing the **data to be displayed** to the user, is created. The **success callback**:

- will locally save the response
- will create the menu
- will show it.

If the menu will contain **elements** that it is possible to **show on the map** they will be added to the map by last function

ParkingSearcher Module Template

- Before adding the logic of the new module, we create the template to be filled with the correct JSON.

```
<div id="parkingMenuHeader" class="panel panel-default" style="position: absolute;right: 0px;left: 0px;border-radius: 0px;">
  <div id="parkingMenuExpandHandler" class="grippyContainer grippyContainer-horizontal" style="text-align: center;">
    <div class="grippy grippy-horizontal"></div>
  </div>
  <div class="panel-heading" style="padding: 0px 10px;height: 52px; border: none;">
    <a class="pull-right" onclick="ParkingSearcher.hide();">
      <i class="glyphicon glyphicon-remove"
        style="float: right; padding-left: 8px; color: #777; line-height: 52px;"></i>
    </a>
    <a id="parkingMenuExpand" class="pull-left" onclick="ParkingSearcher.expandParkingSearcher();">
      <i class="glyphicon glyphicon-plus" style="padding-right: 8px; color: #777; line-height: 52px;"></i>
    </a>
    <a id="parkingMenuCollapse" class="pull-left" onclick="ParkingSearcher.collapseParkingSearcher();">
      <i class="glyphicon glyphicon-minus" style="padding-right: 8px; color: #777; line-height: 52px;"></i>
    </a>
    <b id="parkingMenuHeaderTitle" style="line-height: 52px;color: #333;">
      <script>
        $("#parkingMenuHeaderTitle").html(
          Globalization.labels.parkingMenu.title
        )
      </script>
    </b>
  </div>
</div>

<div id="parkingMenuInner" class="commonHalfMenuInner">
</div>
```

ParkingMenu.mst.html

This default template will **simply show a menu with a header and body empty. Must have the same name as the string entered as the third parameter in the call**

```
ViewManager.render (
  ParkingSearcher.results,
  "#" + ParkingSearcher.idMenu,
  "ParkingMenu");
```


ParkingSearcher Module Template

- Before adding the logic of the new module, we create the template to be filled with the correct JSON.

```
<div id="parkingMenuHeader" class="panel panel-default" style="position: absolute;right: 0px;left: 0px;border-radius: 0px;">
  <div id="parkingMenuExpandHandler" class="grippyContainer grippyContainer-horizontal" style="text-align: center;">
    <div class="grippy grippy-horizontal"></div>
  </div>
  <div class="panel-heading" style="padding: 0px 10px;height: 52px; border: none;">
    <a class="pull-right" onclick="ParkingSearcher.hide();">
      <i class="glyphicon glyphicon-remove"
        style="float: right; padding-left: 8px; color: #777; line-height: 52px;"></i>
    </a>
    <a id="parkingMenuExpand" class="pull-left" onclick="ParkingSearcher.expandParkingSearcher();">
      <i class="glyphicon glyphicon-plus" style="padding-right: 8px; color: #777; line-height: 52px;"></i>
    </a>
    <a id="parkingMenuCollapse" class="pull-left" onclick="ParkingSearcher.collapseParkingSearcher();">
      <i class="glyphicon glyphicon-minus" style="padding-right: 8px; color: #777; line-height: 52px;"></i>
    </a>
    <b id="parkingMenuHeaderTitle" style="line-height: 52px;color: #333;">
      <script>
        $( "#parkingMenuHeaderTitle" ).html(
          Globalization.label: parkingMenu.title
        );
      </script>
    </b>
  </div>
</div>
<div id="parkingMenuInner" class="commonHalfMenuInner">
</div>
```

This template will be saved in the folder called «**templates**».

To add a title to the header we should add this item to all files labels.*. Json

```
{
  "principalMenu": {
    "moduleParkingSearcher": "Lista Parcheggi"
  },
  "parkingMenu": {
    "title": "Parcheggi"
  }
}
```

templates/ParkingMenu.mst.html

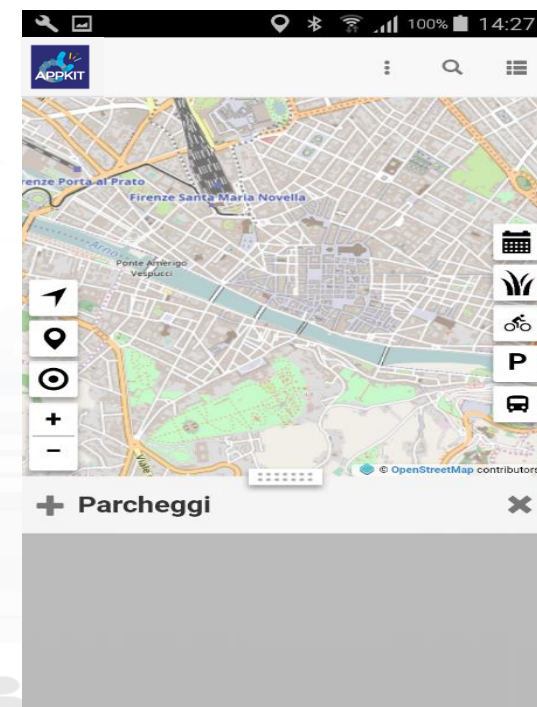
ParkingSearcher Module Template

- Before adding the logic of the new module, we create the template to be filled with the correct JSON.

```
<div id="parkingMenuHeader" class="panel panel-default" style="position: absolute;right: 0px;left: 0px;border-radius: 0px;">
  <div id="parkingMenuExpandHandler" class="grippyContainer grippyContainer-horizontal" style="text-align: center;">
    <div class="grippy grippy-horizontal"></div>
  </div>
  <div class="panel-heading" style="padding: 0px 10px;height: 52px; border: none;">
    <a class="pull-right" onclick="ParkingSearcher.hide();">
      <i class="glyphicon glyphicon-remove"
        style="float: right; padding-left: 8px; color: #777; line-height: 52px;"></i>
    </a>
    <a id="parkingMenuExpand" class="pull-left" onclick="ParkingSearcher.expandParkingSearcher();">
      <i class="glyphicon glyphicon-plus" style="padding-right: 8px; color: #777; line-height: 52px;"></i>
    </a>
    <a id="parkingMenuCollapse" class="pull-left" onclick="ParkingSearcher.collapseParkingSearcher();">
      <i class="glyphicon glyphicon-minus" style="padding-right: 8px; color: #777; line-height: 52px;"></i>
    </a>
    <b id="parkingMenuHeaderTitle" style="line-height: 52px;color: #333;">
      <script>
        $("#parkingMenuHeaderTitle").html(
          Globalization.labels.parkingMenu.title
        )
      </script>
    </b>
  </div>
</div>

<div id="parkingMenuInner" class="commonHalfMenuInner">
</div>
```

templates/ParkingMenu.mst.html



Create ParkingSearcher Module

The goal of this example is to create a **new module** that in addition to viewing the list of car parks as is already the case for the button named "Car Park" will **show directly** the **number of free parking lots** for each car park found

In ParkingSearcher.js must be made the logic that **retrieves data** from API describer in previous presentations and creates the **JSON** to fill the **template** and generate the new menu

ParkingSearcher Called API

- The following API returns **the list of parking** that are located at a maximum distance of 300 meters from the location sent. The list is limited to 100 items.

<http://www.disit.org/ServiceMap/api/v1/?>

selection=**43.7778;11.2481**&

categories=**Car_park**&

maxResults=100&

maxDists=0.3&

format=json&

lang=it&

geometry=true

ParkingSearcher Called API

- The returned data are not sufficient to create the final JSON, because these **data are lacking** on the realtime information

```
▼ object {1}
  ▼ Services {3}
    fullCount : 5
    type : FeatureCollection
    ▼ features [5]
      ▼ 0 {4}
        ▼ geometry {2}
          type : Point
          ► coordinates [2]
          type : Feature
          ▼ properties {7}
            name : Garage La Stazione Spa
            tipo : Parcheggio_auto
            typeLabel : Parcheggio auto
            serviceType : TransferServiceAndRenting_Car_park
            hasGeometry : ☐ false
            serviceUri : http://www.disit.org/km4city/resource/RT04801702315PO
            multimedia : {value}
          id : 1
        ► 1 {4}
```

There are data from **all car parks nearby**, but there are **few properties** that are received

ParkingSearcher Called API

- The following API which returns all information relating to a single service

[http://www.disit.org/ServiceMap/api/v1/?](http://www.disit.org/ServiceMap/api/v1/?serviceUri=http://www.disit.org/km4city/resource/RT04801702315PO&format=json&lang=it)

[serviceUri=http://www.disit.org/km4city/resource/RT04801702315PO&
format=json&
lang=it](http://www.disit.org/km4city/resource/RT04801702315PO&format=json&lang=it)

ParkingSearcher Called API

- The returned data are not sufficient to create the final JSON, because these data are **relative to only one car park**

```
▼ object {2}
  ▼ Service {2}
    type : FeatureCollection
    ▼ features [1]
      ▼ 0 {4}
        ► geometry {2}
          type : Feature
          ► properties {26}
            id : 1
        ► realtime {2}
          ► head {2}
          ▼ results {1}
            ▼ bindings [1]
              ▼ 0 {6}
                ► capacity {1}
                ▼ freeParkingLots {1}
                  value : 282
                ► occupiedParkingLots {1}
                ► occupancy {1}
                ► status {1}
                ► updating {1}
```

There are data from **one car parks nearby**, but there are **many properties** that are received

ParkingSearcher Module Logic

- The idea is to **call the first API that returns the complete list** of nearby car park, and for each car park in the list **call the second API that returns detailed information** with the number of free parking lots

ParkingSearcher Module Logic

- The first API can be call in the app with the following functions

```
search: function(){  
    var parkingQuery = QueryManager.createCategoriesQuery(['Car_park'], SearchManager.searchCenter, "user");  
    APIClient.executeQuery(parkingQuery, ParkingSearcher.searchInformationForEachFeature, ParkingSearcher.errorQuery);  
},
```

<http://www.disit.org/ServiceMap/api/v1/?>

selection=43.7778;11.2481&

categories=Car_park&

maxResults=100&

maxDists=0.3&

format=json&

lang=it&

geometry=true

The **first function** creates the string that contains the **parameters** from “?” to the end.

The **second function** adds the URL of the API and makes the call. When the data has been received calls the error or success callback.

ParkingSearcher Module Logic

- The second API can be call in the app with the following functions

```
searchInformationForEachFeature(response) {  
  for (var category in response) {  
    if (response[category].features.length != 0) {  
      ParkingSearcher.responseLength = response[category].features.length;  
      ParkingSearcher.temporaryResponse = {  
        "Results": {  
          "features": [],  
          "fullCount": ParkingSearcher.responseLength,  
          "type": "FeatureCollection",  
        }  
      };  
      Loading.showAutoSearchLoading();  
      for (var i = 0; i < response[category].features.length; i++) {  
        var serviceQuery = QueryManager.createServiceQuery(response[category].features[i].properties.serviceUri, "app");  
        APIClient.executeQueryWithoutAlert(serviceQuery,  
          ParkingSearcher.mergeResults,  
          ParkingSearcher.decrementAndCheckRetrieved);  
      }  
      SearchManager.startAutoSearch(ParkingSearcher.varName);  
    }  
  }  
}
```

For each car park listed is **called the API that returns details.**

If there is **no car park** in the list is called a function which **doubles the radius** of the search area **until at least one car park is in the list** or the radius is greater than 200 km

ParkingSearcher Module Logic

- The number of free parking lots is copied **from realtime object in the properties** to make writing the template easier. Is also added as a property a string that identifies the **text color** based on the number of free parking lots

```
mergeResults: function (response) {  
  for (var category in response) {  
    if (response[category].features != null) {  
      if (response[category].features.length != 0) {  
        if (response.realtime != null) {  
          if (response.realtime.results != null) {  
            if (response.realtime.results.bindings[0] != null) {  
              if (response.realtime.results.bindings[0].freeParkingLots != null) {  
                response[category].features[0].properties.freeParkingLots = response.realtime.results.bindings[0].freeParkingLots.value;  
                if (response[category].features[0].properties.freeParkingLots > 20) {  
                  response[category].features[0].properties.freeParkingLotsColor = "green";  
                } else if (response[category].features[0].properties.freeParkingLots > 0) {  
                  response[category].features[0].properties.freeParkingLotsColor = "orange";  
                } else {  
                  response[category].features[0].properties.freeParkingLotsColor = "red";  
                }  
              }  
            }  
          }  
        }  
      }  
    }  
    ParkingSearcher.temporaryResponse["Results"].features.push(response[category].features[0]);  
  }  
}  
  
ParkingSearcher.decrementAndCheckRetrieved();  
},  
  
decrementAndCheckRetrieved: function(){  
  ParkingSearcher.responseLength--;  
  
  if (ParkingSearcher.responseLength == 0) {  
    ParkingSearcher.successQuery(ParkingSearcher.temporaryResponse);  
    Loading.hideAutoSearchLoading();  
  }  
},  
},
```

This function controls how many calls have already returned the details or returned error.

ParkingSearcher Module Logic

```
successQuery: function (response) {  
    var responseObject = response;  
  
    if (SearchManager.typeOfSearchCenter == "selectedServiceMarker") {  
        MapManager.searchOnSelectedServiceMarker = true;  
    }  
    for (var i = 0; i < responseObject["Results"].features.length; i++) {  
        responseObject["Results"].features[i].id = i;  
        Utility.enrichService(responseObject["Results"].features[i], i);  
    }  
    if (responseObject["Results"].features[0].properties.distanceFromSearchCenter != null) {  
        responseObject["Results"].features.sort(function (a, b) {  
            return a.properties.distanceFromSearchCenter - b.properties.distanceFromSearchCenter  
        });  
    } else {  
        responseObject["Results"].features.sort(function (a, b) {  
            return a.properties.distanceFromGPS - b.properties.distanceFromGPS  
        });  
    }  
  
    ParkingSearcher.results = responseObject["Results"];  
    ParkingSearcher.refreshMenu();  
    ParkingSearcher.show();  
    MapManager.addGeoJSONLayer(responseObject);  
    ParkingSearcher.resetSearch();  
},
```

This is the **function** that receives the **end JSON** and shows it to the user, by creating the marker on the map and **populating** the **list** through the **template**.

The **JSON** is **enriched** with additional information such as **distance from GPS** or from a manual search and **list** is **sorted** according to these values.

ParkingSearcher Module Template

- This is the **final template** that allows you to show the user a list of car parks in its vicinity with an indication of the number of free parking lots

```
<!-- {{#features}} {{#properties}}-->
<div class="panel panel-default" style="margin: 0px 5px 10px 5px; color: #000; text-decoration: none;" onclick="InfoManager.showInfoAboutOneMarker('{{#properties}}
<div class="panel-body card-2" style="position: relative">
  {{#distanceFromGPS}}
  {{#freeParkingLots}}<b style="position: absolute; bottom: 0px; right: 10px; font-size: 24px; color: {{#freeParkingLotsColor}}">
    {{#freeParkingLots}}</b>{{/freeParkingLots}}
  {{/freeParkingLots}}
  {{#imageThumb}}{{/imageThumb}}
  
  <b style="text-align: center">{{#unescapeHtml}}<b style="text-align: center">{{#unescapeHtml}}</b>
  {{#typeLabel}}<br><b id="parkingMenuTextLabel{{#identifier}}<b id="parkingMenuTextLabel{{#identifier}}</b>
  <script>$("#parkingMenuTextLabel{{#identifier}}").html(Globalization.labels.infoMenu.type)</script>
  </b>{{#typeLabel}}</b>{{/typeLabel}}
  {{#distanceFromSearchCenter}}<br><b id="parkingMenuTextSearchDistanceFromSearchCenter{{#identifier}}<b id="parkingMenuTextSearchDistanceFromSearchCenter{{#identifier}}</b>
  <script>$("#parkingMenuTextSearchDistanceFromSearchCenter{{#identifier}}").html(Globalization.labels.textSearchMenu.distanceFromSearchCenter)</script>
  </b>{{#distanceFromSearchCenter}} m{{/distanceFromSearchCenter}}
  {{#distanceFromGPS}}<br><b id="parkingMenuTextSearchDistanceFromGPS{{#identifier}}<b id="parkingMenuTextSearchDistanceFromGPS{{#identifier}}</b>
  <script>$("#parkingMenuTextSearchDistanceFromGPS{{#identifier}}").html(Globalization.labels.textSearchMenu.distanceFromGPS)</script>
  </b>{{#distanceFromGPS}} m{{/distanceFromGPS}}</b>{{/properties}}
</div>
</div>
<!-- {{/features}} -->
```


ParkingSearcher in main menu

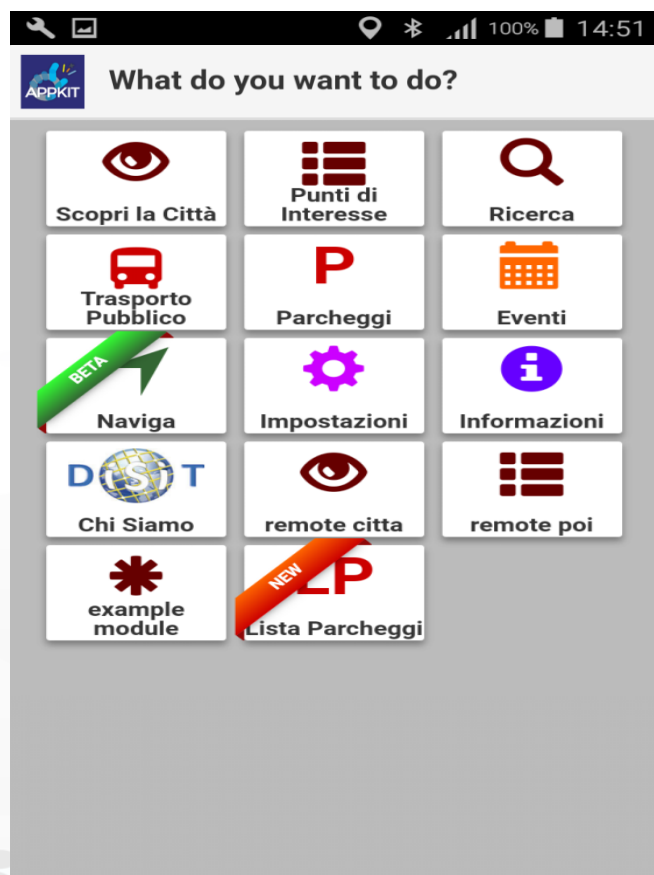
- Final version of the button with call to module logic

```
{
  "callback": "PrincipalMenu.hide(); MapManager.centerMapOnGps() SearchManager.search('ParkingSearcher');",
  "iconId": "",
  "iconClass": "",
  "iconFontSize": "",
  "iconColor": "",
  "imgSrc": "",
  "imgHeight": "",
  "text": "LP",
  "textFontSize": "38px",
  "textColor": "#CC0000",
  "captionId": "principalMenuParkingSearcher",
  "captionTextId": "moduleParkingSearcher",
  "step": "",
  "stepId": "",
  "ribbon": true,
  "ribbonId": "",
  "ribbonStyle": "background: #CC0000;background: linear-gradient(#FF6600 0%, #CC0000 100%);",
  "ribbonText": "NEW",
  "removed": false,
  "index": 0
}
```

The search function of the variable SearchManager **asks the user where want search** (GPS, Manual or Last Service) and then call the **search function** of the variable which is passed as string

parkingSearcher.principalMenu.json

ParkingSearcher Module Finished



For the creation of the app through the modules it is necessary to compile it with grunt before doing it with Cordova



Gruntfile

Inside the root folder of the Sii Mobility App Kit there is a gruntfile.js which will merge the js and json files as described inside it. Inside the root there is also a node_modules directory within which there must necessarily be these folders containing the plugins useful for merging files.

- disit-json-merger
- es6-promise-plugin
- grunt
- grunt-contrib-clean
- grunt-contrib-concat
- grunt-contrib-uglify
- grunt-json-merger

Gruntfile

To compile the files, the grunt command must be launched on the open terminal in the project root

If everything works properly the screen that should appear is as follows. If some package is missing it can be installed with the `npm i packagename` command. If `disit-json-merger` is missing download it from https://github.com/disit/siiMobilityAppKit/tree/master/node_modules/disit-json-merger

```
Running "concat:dist" (concat) task
Running "concat:allTogether" (concat) task

Running "clean:0" (clean) task
>> 1 path cleaned.

Running "disit-json-merger:singleTemplate" (disit-json-merger) task
File "www/js/build/singleTemplate.json" created.

Running "json-merger:ita" (json-merger) task
File "www/js/build/labels.ita.json" created.

Running "json-merger:eng" (json-merger) task
File "www/js/build/labels.eng.json" created.

Running "json-merger:deu" (json-merger) task
File "www/js/build/labels.deu.json" created.

Running "json-merger:esp" (json-merger) task
File "www/js/build/labels.esp.json" created.

Running "json-merger:fra" (json-merger) task
File "www/js/build/labels.fra.json" created.

Done.
```


Gruntfile

If the grunt command was successful as in the image of the previous slide then the command Cordova build android can be launched and if all goes well you will have a screen like the following one

```
:compileDebugSources
:transformClassesWithDexBuilderForDebug
:transformDexArchiveWithExternalLibsDexMergerForDebug
:transformDexArchiveWithDexMergerForDebug
:transformNativeLibsWithMergeJniLibsForDebug
:transformResourcesWithMergeJavaResForDebug
:packageDebug
:assembleDebug
:cdvBuildDebug

BUILD SUCCESSFUL in 1m 54s
44 actionable tasks: 44 executed
Built the following apk(s):
   C:/Users/badii.DISIT/Lavoro/workspace/siiMobilityAppKit/
apk
```


Further readings

- [TC5.16. Exploiting Smart City API for developing Mobile and Web Apps](#)
- [TC5.15. Snap4City Smart City API Collection and overview, real time](#)
- [TC5.17. Search on Services via Smart City API: MicroApplication, Exploiting Micro Applications in HTML5 based on Advanced Smart City API](#)
- [TC5.18. Snap4City API are documented in Swagger, and tested in Postman](#)
- [TC5.19. Using ServiceMap as a Tools for Developing web and mobile apps and micro applications](#)

Useful links

- [US1. Using City Dashboards](#)
- [US2. Using and Creating Snap4City Applications with Dashboards](#)
- [US3. Using and Creating Developer Dashboards, AMMA dashboard, and/or Resource Dashboards](#)
- [US4. Creating City Dashboards and related Event Monitoring and Actions](#)
- [US5. Discovering City Services Exploiting Knowledge Base via ServiceMap](#)
- [US6. Developing and using processes for data transformation](#)
- [US7. Data Analytics and related integration aspects](#)
- [US8. Using the Living Lab Support tools](#)
- [US9. Creating Snap4City IOT Applications, different formats, protocols, brokers, communications](#)
- [US10. Using and Managing the Scalable Snap4City Infrastructure](#)
- [US11. Using tools/services of a secure and privacy respectfully solution](#)

Former Documentation

- **Documentation Smart City API**
 - <http://www.disit.org/6991>
- **Ontology and Km4City Tools:**
 - <Http://www.km4city.org>
 - <http://www.disit.org/6506> Ontology and documentation
- **Snap4city is Open Source on GitHub as DISIT lab:**
 - <https://github.com/disit>
 - <https://github.com/disit/snap4city> (mobile App kit)

TOP

Acknowledgements

FROM CITY
DASHBOARD TO
APPLICATIONS

DATA GATHERING
AND CITY DATA
KNOWLEDGE
MANAGEMENT

FORGING &
MANAGING OPEN
AND FLEXIBLE WEB
AND MOBILE APPS

IOT APPLICATIONS
VS IOT EDGE
DEVICES

IOT APPLICATIONS,
THE LOGIC AND
THE SMARTNESS

ADVANCED
SMART CITY API,
MICROSERVICES,
SNAP4CITY API

SNAP4CITY
LIVING LAB FOR
COLLABORATIVE
WORK

SNAP4CITY FOR
BEGINNERS

DATA BUSINESS
INTELLIGENCE,
WHAT-IF AND
SIMULATION

SNAP4CITY
ARCHITECTURE AND
ECOSYSTEM. OPENED
TO DEVELOPERS
AND STAKEHOLDERS

TWITTER
VIGILANCE: SOCIAL
MEDIA ANALYSIS

DECISION SUPPORT
SYSTEM AND CITY
RESILIENCE

HOW TO ADOPT
SNAP4CITY, AND
OUR ROADMAP

SNAP4CITY
AND KM4CITY
PROJECTS

SNAP4CITY THE
VIEW OF THE
ADMINISTRATORS

...2022

2021

- CAPELON
- Sweden



- Smart Mobility
- PISA, PUMS
- Living lab



- Smart Tourism
- 6 Pilots
- Data Analytics
- Extended platform

2020



- Industry 4.0



- Smart Health



2019

- Traffic and Mobility Impact on Pollution
- NOX predictions



5G tech
Energy
Industry 4.0
Synoptics

- Mobility Demand / Offer Analytics and Strategy

2018



- User engagement
- Bike Sharing
- Data Analytics ++
- Social Predictions
- OBD2



IOT/IOE

Km4City 1.6.6



H2020



(2017-19)

- IOT/IOE, IOT App
- Living Lab
- Maker Support
- IOT Edge
- Smart City IOT
- GDPR,
- Privacy & Security



- Smart Waste

Km4City 1.6.4



- Origin-Destination and trajectories
- Traffic Reconstruction
- Offer Analysis
- OBU, smart devices



GHOST SIR

- Sardinia Region Smart City Strategies and plan



GREEN IMPACT

- Industry 4.0
- Critical Plant
- Monitoring

2017



REPLICATE

- Smart Energy
- Sustainable Mobility
- Control Room
- Dashboard

Km4City 1.6.2



SII-MOBILITY SCN

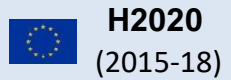
- Infomobility
- Mobile App
- Routing
- Multimodality

2016



Km4City 1.5

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



H2020 (2015-18)

Km4City Ontology 1.1

- Tuscany, Road Graph
- Mobility
- culture, tourism
- Events
- Parking
- Services
- Linked open graph

2014

- Weather Forecast
- Real Time Wi-Fi
- Entertainment
- LOD

- Twitter Vigilance
- Social Media Analytics, Sentiment Analysis

Km4City 1.4

2015



DISIT lab roadmap vs model and tools' usage

Main running instances

SELECT
for Cities



- Sii-Mobility → mobility and transport, sustainability
- REPLICATE → ICT, smart City Control room, Energy, IOT
- RESOLUTE → Resilience, ICT, Big Data
- GHOST → Strategies, smart city
- TRAFAIR → Environment & transport
- MOSAIC → mobility and transport
- WEEE Life → Smart waste, environment
- Smart Garda Lake → Castelnuovo del Garda
- 5G → Industry 4.0 vs SmartCity
- Green Impact → Industry 4.0, Chemical Plant
- SmartBed (laid → smart health
- Green Field Peas (soda) → Industry 4.0, Chemical plant
- MobiMart and PISA Agreement → data aggregation, 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

Acknowledgements

- Thanks to the European Commission for founding. All slides reporting logo of **Snap4City** <https://www.snap4city.org> of **Select4Cities H2020** are representing tools and research founded by European Commission for the **Select4Cities** project. **Select4Cities** has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation Programme (grant agreement n° 688196)
- TRAFair** is a CEF project. All slides reporting logo of TRAFair project are representing tools and research founded by the EC on CEF programme <http://trafair.eu/>
- Thanks to the European Commission for founding. All slides reporting logo of **REPLICATE H2020** are representing tools and research founded by European Commission for the REPLICATE project. **REPLICATE** has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation Programme (grant agreement n° 691735).
- Thanks to the European Commission for founding. All slides reporting logo of **RESOLUTE H2020** are representing tools and research founded by European Commission for the RESOLUTE project. **RESOLUTE** has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation Programme (grant agreement n° 653460).
- Thanks to the MIUR for co-founding and to the University of Florence and companies involved. All slides reporting logo of **Sii-Mobility** are representing tools and research founded by MIUR for the Sii-Mobility SCN MIUR project.
- Km4City** is an open technology and research line of DISIT Lab exploited by a number of projects. Some of the innovative solutions and research issues developed into projects are also compliant and contributing to the Km4City approach and thus are released as open sources and are interoperable, scalable, modular, standard compliant, etc.



DISIT thanks to

Herit Data: Tourism and Mng. <https://herit-data.interreg-med.eu/>

Snap4City: IOT/IOE smart city www.snap4city.org

Trafair: CEF project with several Cities <http://trafair.eu/>

Mosaic: Mobility and transport model

Km4City: <http://www.km4city.org>

REPLICATE H2020, SCC1, EC flagship

<http://replicate-project.eu/>

Sii-Mobility SCN MIUR: <http://www.sii-mobility.org>

Feedback: retail and GDO Big Data analytics

5G with 3G-Wind, Open Fiber, Estra

Coll@bora Social Innovation, MIUR:

<http://www.disit.org/5479>

RESOLUTE H2020, EC:

<http://www.resolute-eu.org>

TRACE-IT, RAISSS, TESYSRAIL, ...

Mobile Emergency:

<http://www.disit.org/5404>



Further readings



<https://www.snap4city.org/108>

- [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](#)

TOP



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



UNIVERSITÀ
DEGLI STUDI
FIRENZE

DINFO
DIPARTIMENTO DI
INGEGNERIA
DELL'INFORMAZIONE

DISIT
DISTRIBUTED SYSTEMS
AND INTERNET
TECHNOLOGIES LAB