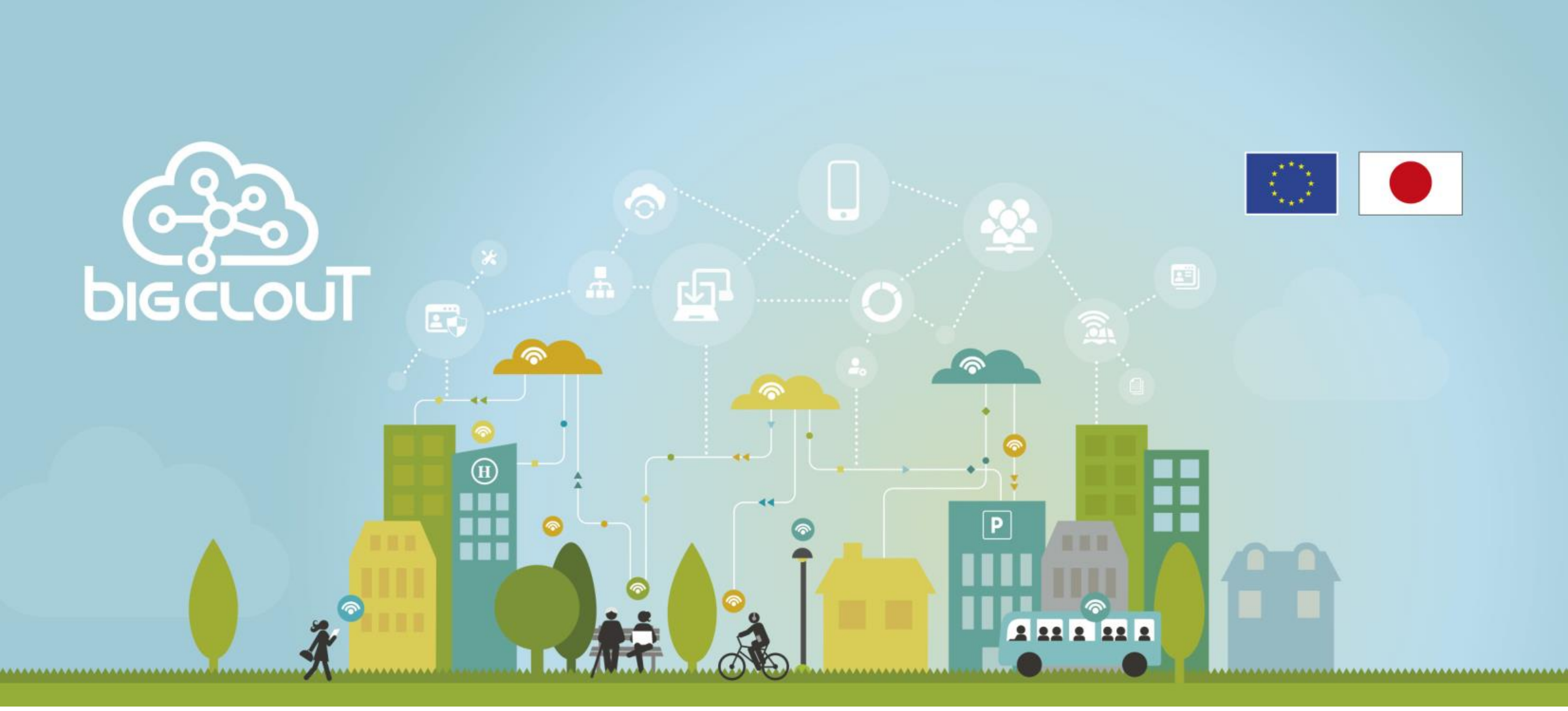


WP4 Pilots and citizen involvement, integration and validation

D4.4 Integrated BigClouT platform

Grant Agreement N°723139
NICT management number: 18301

BIGCLOUT

Big data meeting Cloud and IoT
for empowering the citizen ClouT in smart cities

H2020-EUJ-2016 EU-Japan Joint Call

EU Editor: **CEA** JP Editor**: KEIO** Nature: **Demonstration**

Dissemination: PU

Contractual delivery date: 2019-07-01

Submission Date: 2019-08-02

Co-funded by the EU H2020 GA. 723139 and NICT GA. 18301

ABSTRACT

This deliverable describes the final version of the BigClouT integrated platform, which is composed of various components that have been developed and integrated within the project. The deliverable also provides information about the demonstrators showing the integration work, with screenshots, videos, running instances and links to code repositories, where relevant.

Disclaimer

Revision history

Revision	Date	Description	Author (Organisation)
V0.1	01/07/2019	Initial ToC	Levent Gürgen (CEA), Jin Nakazawa (Keio)
V0.2	11/07/2019	Updated ToC, first inputs	Levent Gürgen (CEA), Jin Nakazawa, (Keio)
V0.3	16/07/2019	First draft for call for contributions	Levent Gürgen (CEA)
V0.4	19/07/2019	Integration of first contributions	Levent Gürgen (CEA)
V0.5	24/07/2019	Integration of second round of contributions	Levent Gürgen (CEA), Jin Nakazawa, (Keio), Takuro Yonezawa (Keio), Orfeas Voutyras (ICCS), Antonis Litke (ICCS), Feng Chen (NII), Toshiyuki Amagasa (TSU), Giuseppe Ciulla (ENG), Hiroyuki Maeomichi (NTTRD), Rodeger Lea (LANC), Matthew Broadbent (LANC)
V0.6	30/07/2019	Integration of final contributions	Levent Gürgen (CEA), Jin Nakazawa (Keio)
V.07	31/07/2019	Draft final version for the internal review	Levent Gürgen (CEA), Jin Nakazawa (Keio)
V0.8	01/08/2019	Integration of the review comments	Levent Gürgen (CEA), Jin Nakazawa (Keio), Orfeas Voutyras (ICCS), Feng Chen (NII),
V1.0	02/08/2019	Submission of the final version	Levent Gürgen (CEA)

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

This deliverable describes the final integrated BigClouT platform, which provides a modular framework to build customised solutions for today's smart city challenges including concerns in interoperability and big data. BigClouT is composed of various components developed by its partners, each one contributing to provide BigClouT main features that can be summarised as follows:

1. BigClouT platform facilitates rapid and flexible collection of a variety of city data (citizens, sensors, web pages, legacy platforms, ...)
2. BigClouT provides high level programming tools for rapid prototyping of smart city applications
3. BigClouT provides easy-to-use tools to extract value from the raw city data
4. BigClouT maintains a data lake with real-life data useful for building and experimenting on city and citizen services
5. BigClouT is based on a modular architecture that can be instantiated and customised according to specific city needs

The BigClouT project is a three-year project, where building the BigClouT platform and tools has been divided into four main phases (see Section 2 for more details):

1. Use cases, requirements analysis and architecture building

BigClouT project has an important advantage of having direct links with cities, either as partners (such as Grenoble Métropole) or connections via local partners (such as Bristol is Open, Tsukuba University or Keio University). This helped in identifying real life use cases and requirements (see Deliverable 1.2 [2]). With those elements as input, within this first phase of the project, the BigClouT architecture and main necessary functional blocks have been identified. Section 2.1 summarizes the overall architecture of BigClouT.

2. Development of functional components

The second phase was the development of the functional components that address the identified needs. The second year's focus was set particularly on this development work. 13 components have been developed/adapted for the specific needs of the project. The main developed components have been presented in Section 2.2.

3. Integration

Once the components were developed, the next step was to integrate and make them work together for common objectives. An important number of integration work has been performed to satisfy the needs of various technical use cases and trial cases. Section 2.3 summarizes this integration work.

4. Deployments and validation in field trials

Based on the identified needs of each trial, the integration work has been prioritized and finalised for the final deployments. Section 3 provides an introduction on how BigClouT platform has been used by various trials. However, detailed information about each trial is given in Deliverable 4.5 [16].

This deliverable is a "demonstrator" type of deliverable, the objective of which is to show the work of integration of project partners' achievements. Technical details of the demonstrated components are given in project deliverables such as D1.1 [1], D1.2 [2], D1.3 [3], D1.4 [4], D2.1 [5], D2.2 [6], D2.3 [7], D2.4 [8], D2.5 [9], D2.6 [10], D3.1 [11], D3.2 [12], D3.3 [13], D3.4 [14], D4.3 [15] and D4.5 [16]. The deliverables are available for download at the project website.

2. BIGCLOUT EVOLUTION FROM 1ST TO THE FINAL DAY

2.1 First year focus – architecture

One of the main focus points of the first year had been to build a reference architecture for the BigClouT platform and tools. Following the analysis of the trial use cases and requirements, the BigClouT architecture was built in the Workpackage 1 of the project. While the Deliverable 1.4 provided the high level BigClouT architecture, as well as detailed description of its different functional blocks, the figure below illustrates a simplified version of the architecture focusing on main functional layers.

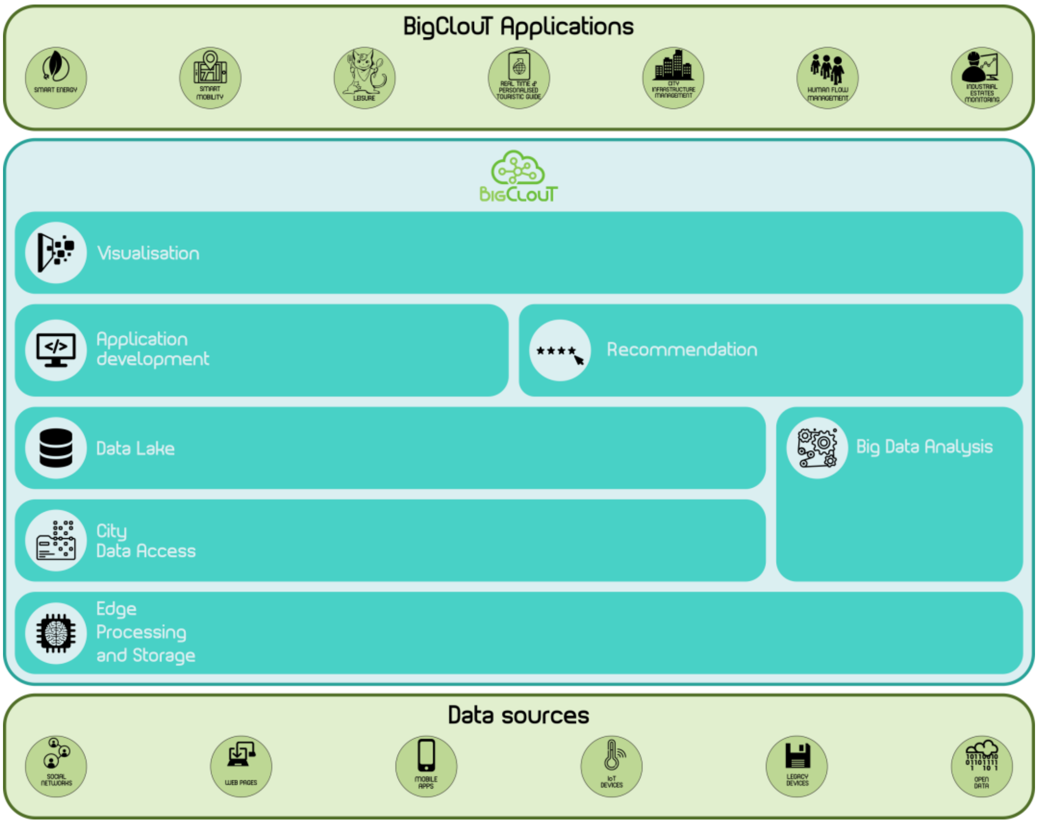


FIGURE 1 BIGCLOUT OVERALL ARCHITECTURE

Each layer is briefly described in the following:

Data sources layer

This layer is about all types of data sources that we can encounter in city environments, such as IoT devices, legacy platforms, social networks, mobile apps, web pages, open data, etc. BigClouT platform is a middleware layer between data sources and applications, providing various functionalities such as data collection, processing, analysis, recommendation and visualisation.

BigClouT functional blocks

- **Edge Processing and Storage**
  This block is in charge of managing edge storage and edge processing capabilities provided by the BigClouT platform. The edge processing (resp. storage) is about *distributed processing* (storage) of data close to its source either for performance reasons or data privacy, so that transferring data to the cloud is avoided For instance, image processing on the edge in the

BigClouT road condition monitoring scenario by the cameras deployed at the garbage trucks is a good example of processing on the edge for privacy reasons.

- **City Data Access**
  This is the main access point to the data produced by the data sources (either raw data or processed data at the edge). Homogenisation of data coming from a variety of heterogeneous sources is done at this level. Common APIs and data models allow developers to easily access the data without any particular knowledge on the underlying protocols and data models of each source.

- **Data Lake**
  This is the main storage entity providing facilities such as data catalogue and data look-up. The data lake mainly contains static data, which can be either raw or processed data generated by big data analysis tools.

- **Big Data Analysis**
  This is the layer where the processing in the cloud takes place. The tools take as input a large quantity of data and then apply aggregation functions either on static or streaming data and provides outputs either in batches or in a continuous manner.

- **Recommendation Service**
  The big data analysis combined with a reasoning engine allows the platform providing useful recommendations for different situations such as recommendations for efficient energy usage, mobility paths taking into account the traffic situation and pollution levels, etc.

- **Application Development**
  Smart city is a multi-disciplinary domain in which various stakeholders can get involved in contributing to building the city of the future. Despite limited technical expertise, those stakeholders should be able to contribute to this vision. The easy-to-use tools for application development from BigClouT aims at addressing this need.

- **Visualisation**
  Visualisation tools are essential in quickly gaining understandings of the situations in city environments. Main required properties for the visualisation tools are easiness of use, interoperability of heterogeneous data sources and extensibility with new visual widgets.

BigClouT applications

BigClouT project has been addressing a vast array of applications coming from 4 project pilot cities: Grenoble, Bristol, Tsukuba and Fujisawa. The main use cases domains include: smart energy, smart mobility, leisure, real-time personalised touristic guide, city infrastructure management, human flow management, industrial estates monitoring. More information about the BigClouT trial cases can be found in the Deliverable D4.5 [16].

2.2 Second year results – individual components

The second year of the project focused on further developing the necessary components to respond to the functional needs identified during the requirements analysis and architecture construction work. The consorrtium partners have developed necessary functions either from scratch or by extending existing assets that they brought into the project. The following list provides a brief description of each of the 13 assets that have been developed/extended in the BigClouT project:

FIGURE 2 BIGCLOUT COMPONENT ASSETS

1. **KNOWAGE** (ENG)

KNOWAGE is an open source business intelligence suite providing different analytical tools to create reports, charts, graphs and interactive cockpits, in order to visualise data on maps, to perform multidimensional analysis, to extract knowledge, etc. KNOWAGE supports a wide range of data sources, structured databases, NoSQL databases, and also different tools and platforms. The BigClouT project contributed in improving its performance and capabilities. More specifically, widgets to represent charts and graphs have been refined and its capabilities to interact with external data sources, such as REST APIs, have been improved. More information about KNOWAGE can be found at https://www.knowage-suite.com/site/home/.

2. **Recommendation Service** (ICCS)

This service provides personalised recommendations to end users/citizens in several application settings such as energy consumption, transportation management, tourism services, air and sound pollution management and many others. During BigClouT, the Recommendation Service was developed based on a distributed Neo4j graph database, Node-

RED and exposed RESTful APIs, enabling easier integration with other tools and services. The API manages the users' requests for recommendations and the connection with data sources, and handles user generated data as well. The code is open source and can be found at https://github.com/RecommendationIoT with documentation and additional information.

3. **StreamingCube** (TSU)

StreamingCube is a unified framework for stream processing and OLAP analysis. StreamingCube supports both continuous queries (CQs) and OLAP queries. Due to BigClouT, unique feature of CQs in StreamingCube was developed: CQs now can reference lattice vertices generated by the **cubify** operator, which enables CQs to generate output streams based on precomputed aggregates. StreamingCube is distributed by Kitagawa & Amagasa Data Engineering Laboratory, University of Tsukuba under the Apache License, Version 2.0. For more information available at: http://www.streamspinner.org/streamingcube/documentation.html

4. **BigClouT data lake** (LANC)

The BigClouT Data Lake is a shared repository that is used to manage both static and real-time data from BigClouT partners. Based on the popular CKAN open data platform, the BigClouT project enabled its extension with a real-time capability allowing it to provide high performance queries over real time (streaming) data from city sources. Currently managing approximately 120GB of BigClouT city data, it has over 100 data resources stored in a variety of formats (JSON, csv, txt, pdf, etc.) which can be queried directly via a standardised API, via http queries or via an SQL based query language. The data lake is available at the link: http://bigclout.lancaster.ac.uk/dataset

5. **sensiNact** (CEA)

sensiNact is an open IoT platform dedicated to the smart city domain. It provides support for various IoT protocols and platforms to provide a homogenous access over heterogeneous city data sources. The BigClouT project enabled the development and validation support for additional data platforms and IoT protocols, thus increasing sensiNact's readiness level. More information about sensiNact can be found at https://projects.eclipse.org/projects/technology.sensinact. sensiNact is an open source project and its code is accessible at https://git.eclipse.org/c/sensinact/org.eclipse.sensinact.gateway.git

6. **sensiNact Studio** (CEA)

sensiNact Studio is a visual tool to monitor and manage data/action providers (e.g. IoT devices, data sources) and build rule-based applications exploiting resources (e.g. sensors, actuators) exposed by those providers. The BigClouT project enabled its integration with a verifier mechanism to detect and resolve conflictual accesses to resources by concurrent applications. More information about sensiNact Studio can be found at https://projects.eclipse.org/projects/technology.sensinact. sensiNact Studio is an open source project and its code is accessible at https://git.eclipse.org/c/sensinact/org.eclipse.sensinact.studio.git

7. **SoXFire** (Keio)

SOXFire is a multi-community city-wide sensor network for sharing big, social, sensor data in smart cities. The goal of SOXFire is to provide practical distributed and federated infrastructure for IoT sensor data sharing among various users/organisations in a way that is scalable, extensible, easy to use and secure with preserving privacy. SOXFire supports not only access to physical IoT sensors but also crowd sensing and SNS/Web sensing where city employees, citizens and WEB developers contribute in a different ways, while using unified APIs. The BigClouT project enabled the tool's deployment to other cities (going international) and enhancement of flexibilities as a common platform element. SOXFire is open source and its code can be found at http://sox.ht.sfc.keio.ac.jp/.

8. **DeepOnEdge** (Keio)

DeepOnEdge (DoE) is based on edge computing and deep convolutional neural networks (a type of deep neural networks). Within BigClouT, a convolutional neural network (CNN) had been trained on labelled images on a GPU server. At inference time, DoE is loaded on an edge computer and gives as output a discrete probability distribution, assigning each image a likelihood value. For example, the detection of damaged white lines on the roads of Fujisawa city has been demonstrated. The BigClouT project offered the opportunity of the tools deployment to additional cities in Japan. DoE is not an open source yet, the possibility of providing DoE as open source is investigated.

9. **Edge Storage** (ENG)

The Edge Storage is composed by a Cloud Node (Master), and an arbitrary number of geographically distributed Edge Nodes (Slaves). It allows storing a potentially unlimited amount of data through the Cloud, reducing the latency by using the locally deployed Edge Nodes. All the Nodes offer standard CDMI endpoints and data uploaded on a certain Node can be downloaded from whatever Node of the Storage is needed. Starting from the Cloud Storage produced in the ClouT project, BigClouT project enabled the development of the edge components from scratch, including the functionalities related to the Distributed Edge Nodes. The code is open source (Apache 2 licence) and is publicly available on ENG GitLab (https://production.eng.it/gitlab/bigclout/cdmi-storage).

10. **D-NR platform** (LANC)

The D-NR platform is a distributed edge processing platform that supports automatic replication and distribution of cloud and edge processing modules with built in load balancing. The platform utilises the popular Node-RED visual programming tool (see studio below) allowing developers to use a drag and drop metaphor to build smart city applications and services. The BigClouT project enabled the development of an edge processing capability as part of the D-NR platform and the integration of the platform with D-NR studio, something that allowed developers to use an easy visual annotation tool to apply constraints to smart city applications that are then used by the underlying D-NR platform to drive edge processing and load balancing. D-NR is an open source platform. More details can be found at https://github.com/namgk/dnr-editor/blob/master/README.md and the source code can be downloaded from https://github.com/namgk/dnr-editor.

11. **D-NR Studio** (LANC)

D-NR studio is the user interface and development tool for the D-NR edge processing platform and is built on the popular Node-RED visual programming tool. Studio allows developers to rapidly build smart city applications and services using a simple drag and drop metaphor and has a build in capability for building smart city visualisations and dashboards. It has been further augmented with a set of Node-RED modules for easy programming of BigClouT services, e.g. the Data Lake, SOXFire, sensiNact, etc., allowing rapid development of BigClouT applications. As indicated above, it has been extended with a constraint tool allowing developers a simple way to control the distributed edge processing capabilities of D-NR platform. D-NR is an open source platform. More details can be found at https://github.com/namgk/dnr-editor/blob/master/README.md and the source code can be downloaded from: https://github.com/namgk/dnr-editor.

12. **ECA Verifier** (NII)

ECA Verifier is an open source plugin for sensiNact studio and gateway, which implements the functionality of self-aware conflict resolution for service composition. It improves the degree of dependability of the platform by automatically detecting and resolving conflicts among applications developed and deployed in sensiNact. The BigClouT project allowed the demonstration of the feasibility of the verifying mechanism based on ECA and thus improved the verifier's readiness level. More information about the ECA Verifier, including its code, can be found at https://github.com/ForgingAhead/ECA-Verifier-sensinact.

13. **oneM2M implementation** (NTTR&D)

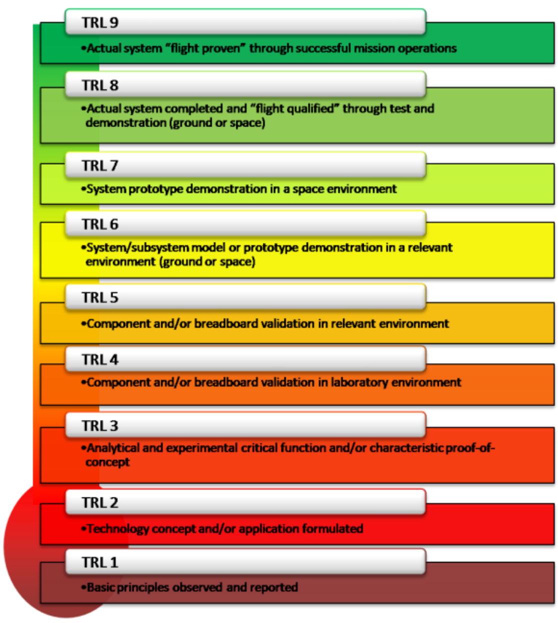
NTT has developed NILONE (NTT Innovation Laboratories' oneM2M platform), a high performance implementation of oneM2M[1] standard, fully compliant to the specifications with high performance. In the BigClouT context, NILONE provides the standardised interface for providing data from a wide range of city data sources. The BigClouT enabled the creation of a standard API for oneM2M application support in both OSGi Alliance[2] and oneM2M[3].

The following table illustrates the Technology Readiness Level of each of the BigClouT assets

TABLE 1 BIGCLOUT ASSETS AND THEIR TRL LEVEL

BigClouT Assets	Initial TRL	Final TRL
BigClouT data lake	8	9
DeepOnEdge	1	6
D-NR platform	3	6
D-NR studio	5	6
ECA verifier	3	4
Edge storage	3	4
Knowage	5	7
oneM2M	4	6
Recommendation service	3	6
SensiNact	5	7
SensiNact Studio	4	6
SOXFire	3	7
StreamingCube	2	6

Tecnology Readiness Level Scale



[1] http://onem2m.org
[2] https://github.com/osgi/design/raw/master/rfcs/rfc0237/rfc-0237-Service-Layer-API-for-oneM2M.pdf
[3] http://ftp.onem2m.org/Meetings/ARC/20181203_ARC38_Kanazawa/ARC-2018-0336-OSGi_Service_Layer_API.DOC).

2.3 Final year focus - integrations

While the 2nd year focused on the development work, the 3rd year's main focus have been the integrations and deployments. A total of about 20 integration work has been done during the last 18 months of the project (See Figure 3).

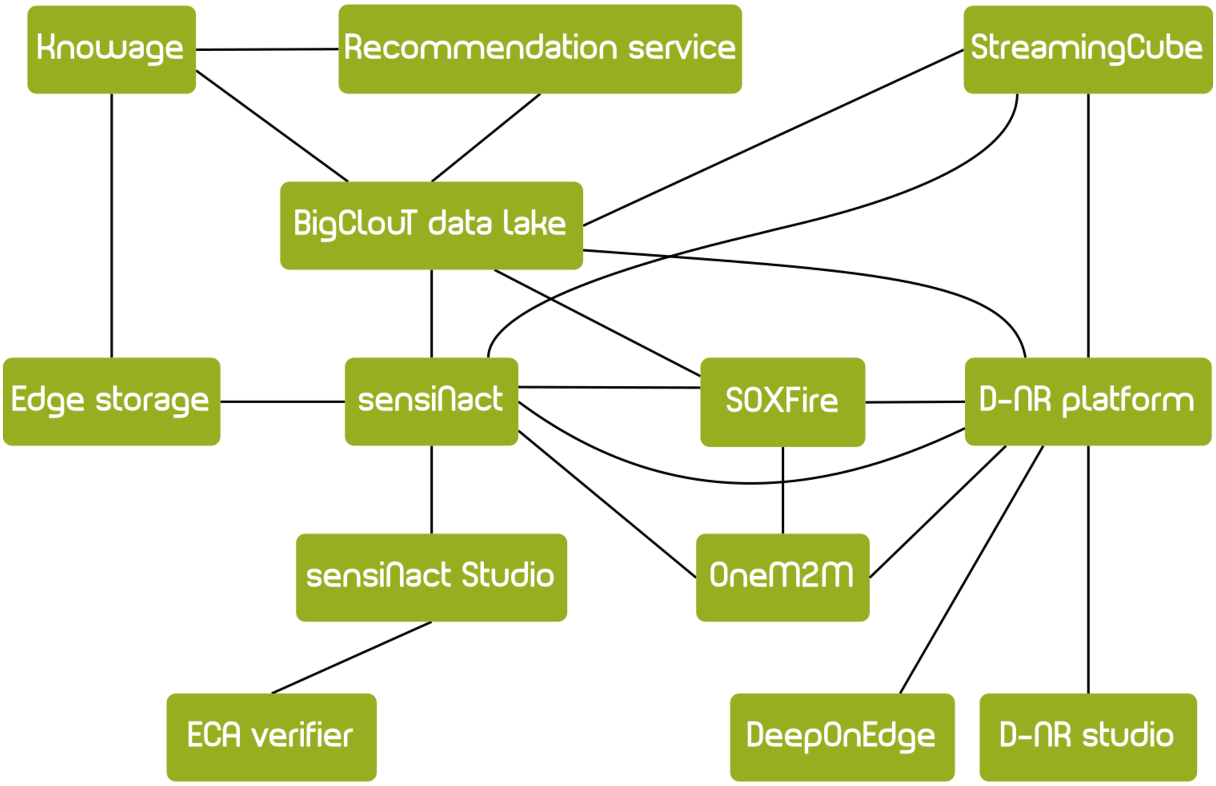


FIGURE 3 INTEGRATIONS AMONG THE BIGCLOUT COMPONENTS

The following subsections describe the integration work and provide, where relevant, screenshots, videos and tutorials about the integration work. It shuld be noted that many of those integrations have been used for the BigClouT trials which are described in the Deliverable 4.5. Thus the integrations will be demonstrated during the project review meeting that will take place on September 19th 2019 in Tokyo.

2.3.1 sensiNact integrations (CEA)

sensiNact is mainly placed at the level of data collection and redistribution. sensiNact's main goal is to facilitate access to a large variety of distributed data sources in city environments. sensiNact is mainly used with its visualisation and programming tool sensiNact Studio. This section summarizes the integration work that has been established between the BigClouT components and the sensiNact platform and its Studio.

2.3.1.1 sensiNact – SoxFire integration (CEA-Keio)

The sensiNact – SoxFire integration allows the sensiNact platform gathering data exposed by the SoxFire platform. A specific sensiNact – SoxFire bridge has been developed providing access to various data in Japan exposed by SoxFire (sensorised garbage trucks, sensorised web pages, minarepo reports, etc.).

The code will soon be released to the Eclipse repository of sensiNact: https://git.eclipse.org/c/sensinact/org.eclipse.sensinact.gateway.git (pending for license validation by Eclipse foundation).

The running instance can be accessed by using the following link. Once accessed to the page, click on the "+" button and click on "Add" accepting the default gateway configuration data:

http://sensinact.ddns.net/studio-web/

A screenshot from sensiNact Studio showing the data sources from Japan integrated to sensiNact via SOXFire can be found in the Figure 4.

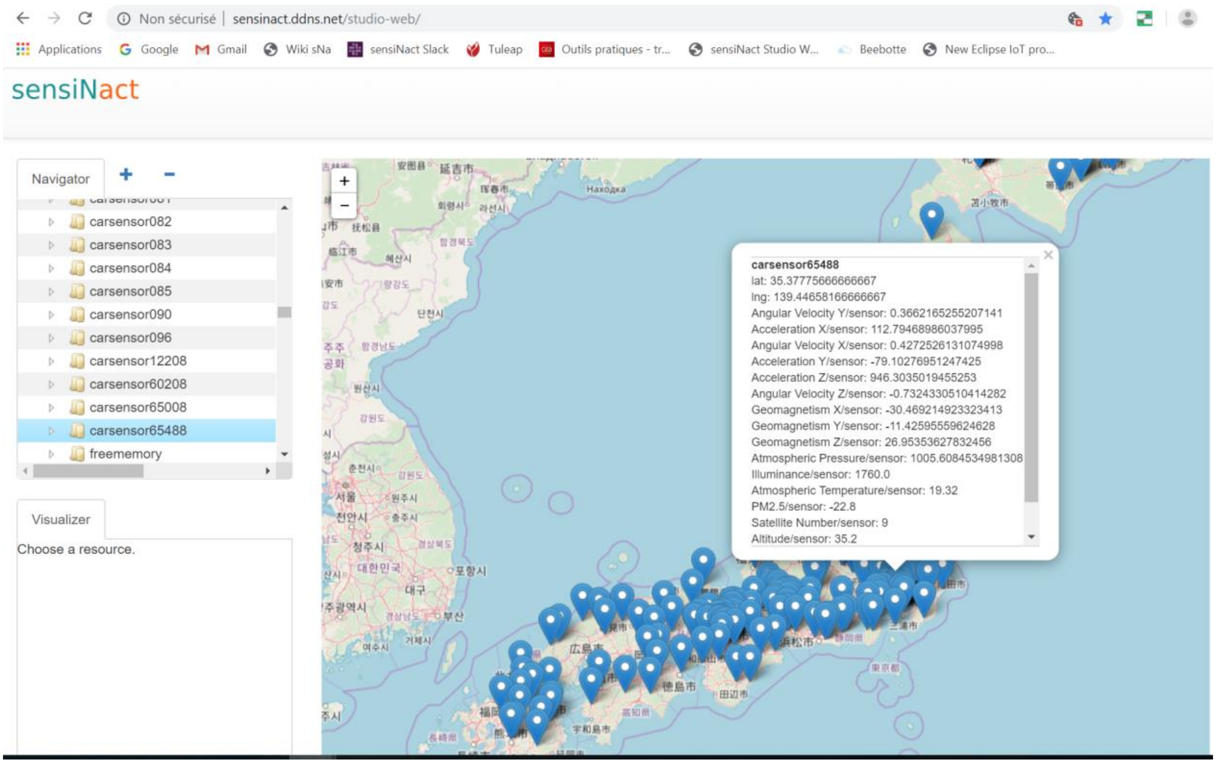


FIGURE 4 DATA FROM JAPAN COLLECTED BY SENSINACT VIA SOXFIRE

2.3.1.2 sensiNact – BigClouT data lake integration (CEA-LANC-ENG-ICCS)

A specific northbound bridge for sending data to the BigClouT data lake has been developed. Data from Inovallée pilot is being sent to the data lake and it is available in the data lake.

http://bigclout.lancaster.ac.uk/dataset/grenoble-inovallee

Innovallée data are being used by KNOWAGE for analysis to respond to various questions such as, which event attracts more attention, which restaurants are more looked for, at what time of the day users use the app, which bus stations they use more, etc. Various visualisation graphs have been provided to Inovallée for the purpose of understanding the behaviours of the Inovallée employees.

Figure 5 below provides some screenshots from the provided dashboards, which is also accessible at the following link:

https://knowage.opsi-lab.it/knowage/public/servlet/AdapterHTTP?ACTION_NAME=EXECUTE_DOCUMENT_ACTION&OBJECT_LABEL=Gre_Usage&TOOLBAR_VISIBLE=true&ORGANIZATION=BigClouT&NEW_SESSION=true&PARAMETERS=day=&day_field_visible_description=

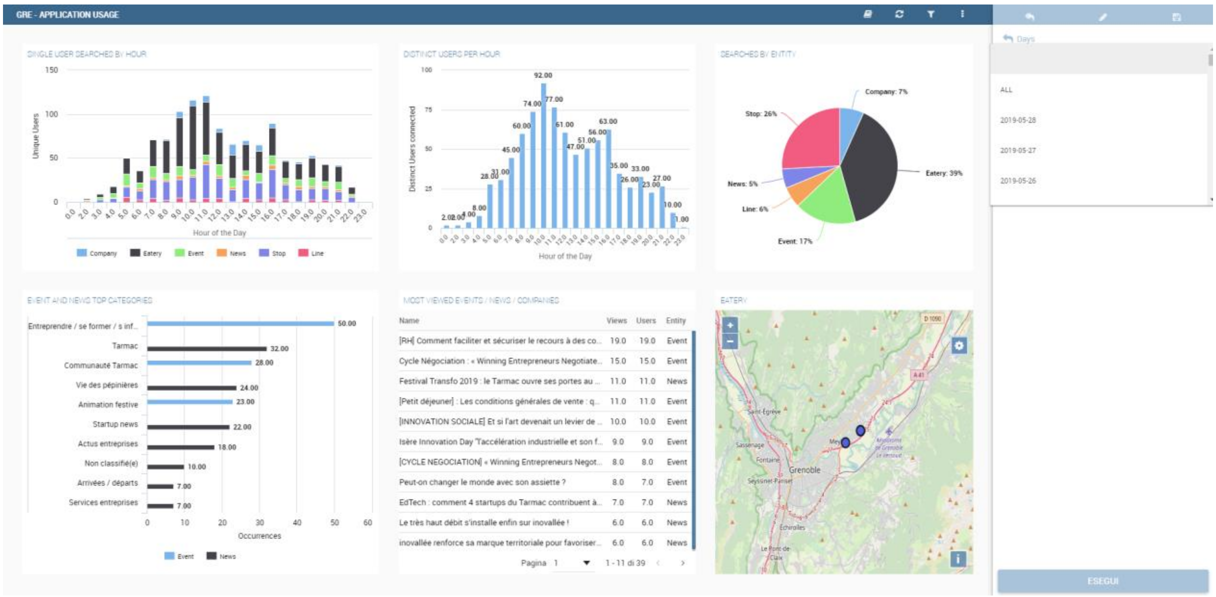


FIGURE 5 KNOWAGE DASHBOARDS FROM INOVALLÉE DATA SENT BY SENSINACT

In addition, another integration work is ongoing for providing a recommendation service for the Inovallée employees to provide them recommendations about after-work or lunch time events, restaurant recommendations or better personalized mobility options.

2.3.1.3 sensiNact – Edge Storage integration (CEA-ENG)

Edge Storage paradigm enables sharinglimited storage capabilities, thus obtaining a distributed storage. Cloud technology, indeed, provides potentially unlimited storage capabilities offered as a Service by hardware located on big data centres. The advantage of the former is latency: A Node, even if it is not a smart device, can be deployed everywhere and in particular close to the place in which data are produced and consumed. This is useful to store limited set of data to be immediately processed on the same place. The combination of the Edge paradigm and Cloud technology combines high capabilities and low latency: these are the features of the Fog Paradigm.

Based on the integration that has been done between the CDMI storage and sensiNact in the preceding ClouT project, this integration has been extended to the distributed CDMI based on the same API. The ultimate goal is having a transparent storage system in which the system automatically handles the transfer, balancing and replication of data among various storage edge nodes and the cloud storage (See Figure 6).

The integration is still ongoing. The code will be uploaded to the Eclipse repository once finalised.

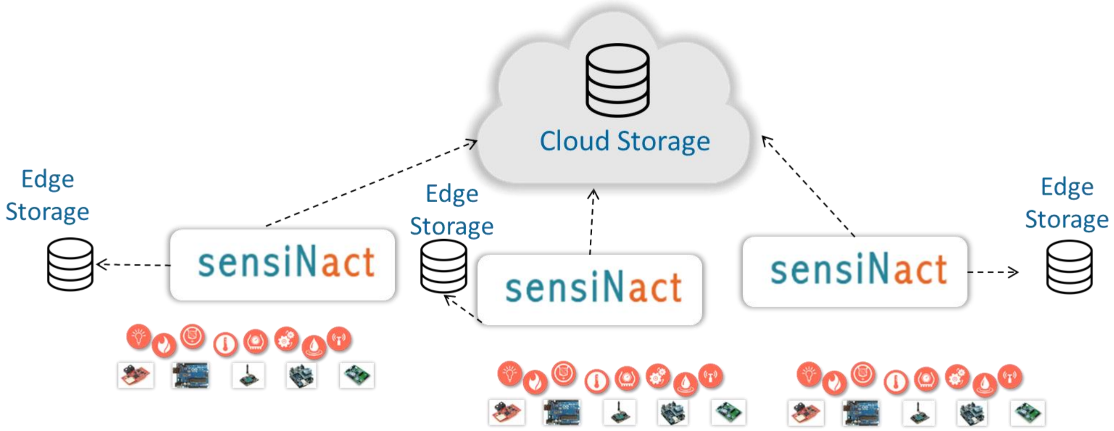


FIGURE 6 EDGE STORAGE INTEGRATION WITH SENSINACT

2.3.1.4 sensiNact - oneM2M integration (CEA-NTTR&D)

sensiNact platform provides a hub of communication to various types of protocols from/to southbound and northbound protocols. Within BigClouT, both southbound and northbound bridges for oneM2M have been developed. The bridges provide the translation between the Eclipse sensiNact data model and the one from oneM2M. The northbound bridge (using MQTT as underlying data transfer protocol) allows anyone accessing the whole data from sensiNact by using oneM2M APIs and oneM2M data model. Similarly, the southbound bridge (using HTTP as underlying data transfer protocol) allows any oneM2M devices to send data to sensiNact, which can be gathered by sensiNact APIs in sensiNact data model. The code is available in an open source repository, allowing others users to deploy it to integrate oneM2M platforms.

Northbound bridge:
https://git.eclipse.org/c/sensinact/org.eclipse.sensinact.gateway.git/tree/platform/northbound/http-onem2m-agent

Southbound bridge:
https://git.eclipse.org/c/sensinact/org.eclipse.sensinact.gateway.git/tree/platform/southbound/http/onem2m-device

2.3.1.5 sensiNact – sensiNact Studio integration (CEA)

sensiNact is an evolving platform continuously including new features, southbound and northbound bridges, support for new tools, etc. sensiNact Studio needs to be updated to be in line with the updates to the sensiNact platform. Latest release of sensiNact Studio thus includes the support for the updates done to the sensiNact platform in this third year (support for websockets, secure connections, data model update, etc.).

Figure 7 shows a screenshot of the Studio integrating data from the Inovallée deployment such as bus stations, restaurants and events in the zone as sensiNact service providers.

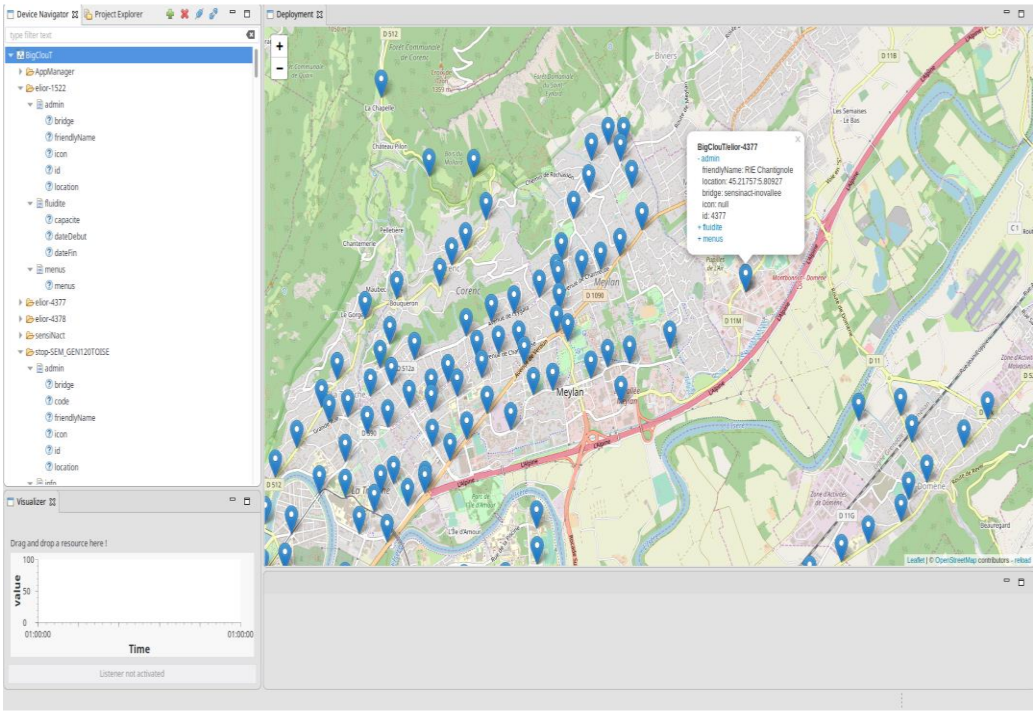


FIGURE 7 SENSINACT STUDIO VISUALISING BUS STOPS, RESTAURANTS, EVENTS AS SENSINACT SERVICE PROVIDERS

The latest version of the implementation of the Studio can be found at https://git.eclipse.org/c/sensinact/org.eclipse.sensinact.studio.git.

The developments related to the Inovallée can be found at the https://git.eclipse.org/c/sensinact/org.eclipse.sensinact.gateway.git/tree/platform/southbound/inovallee.

2.3.1.6 sensiNact Studio – ECA Verifier integration (CEA-NII)

This integration addresses the challenge of conflict detection and resolution among applications developed in the sensiNact Studio and deployed in the sensiNact platform. To implement and demonstrate this use case, an ECA Verifier is developed as a plugin to the sensiNact Studio that is able to automatically detect and resolve conflicts among applications developed by one user. Furthermore, the core functions of the ECA Verifier can also be integrated to the platform to resolve conflicts among applications developed by different users that are deployed in the same platform.

The screenshot below is captured in the sensiNact studio when connecting to the platform, namely the 'localgateway' as shown in Figure 8. Those resources such as 'button', 'slider', and 'light', etc. are simulated devices in the 'localgateway' platform. This demo is an integration effort made between the sensiNact studio and the ECA Verifier which detects and resolves conflicts among applications developed by one user. As a plug-in, the ECA Verifier implements an extension where the Eclipse extension point is defined in the studio. When an IoT application is going to be deployed or un-deployed, the plugin will then be triggered.

More information about the integration work can be found in the Deliverable 2.6 [10].

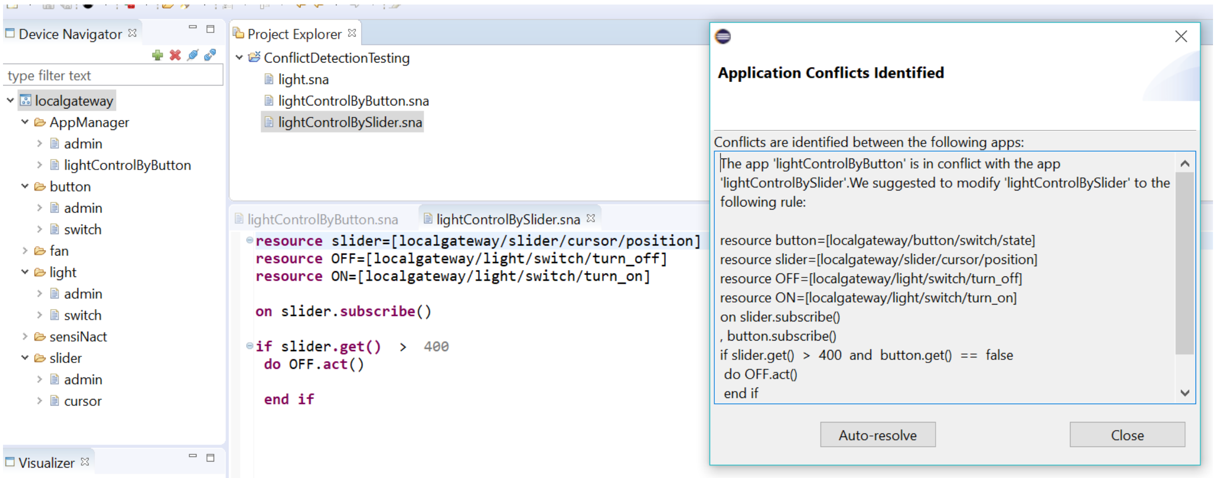


FIGURE 8 APPLICATION CONFLICTS IDENTIFIED DIALOG FOR DEVELOPERS IN SENSINACT STUDIO

2.3.2 SOXFire integrations (Keio)

2.3.2.1 SOXFire – oneM2M – D-NR integration (Keio-LANC-NTTR&D)

SOXFire – oneM2M integration is done in order to demonstrate generic oneM2M applications that can reuse data which are redistributed by the BigClouT platform. For demonstration purposes, two applications have been built, reusing the data collected by garbage trucks. The first one is an application that visualises the trucks and provide a pollution map (Figure 9). The application splits the map into segments and the latest value of the vehicle in the segment is regarded as the value of the segment.

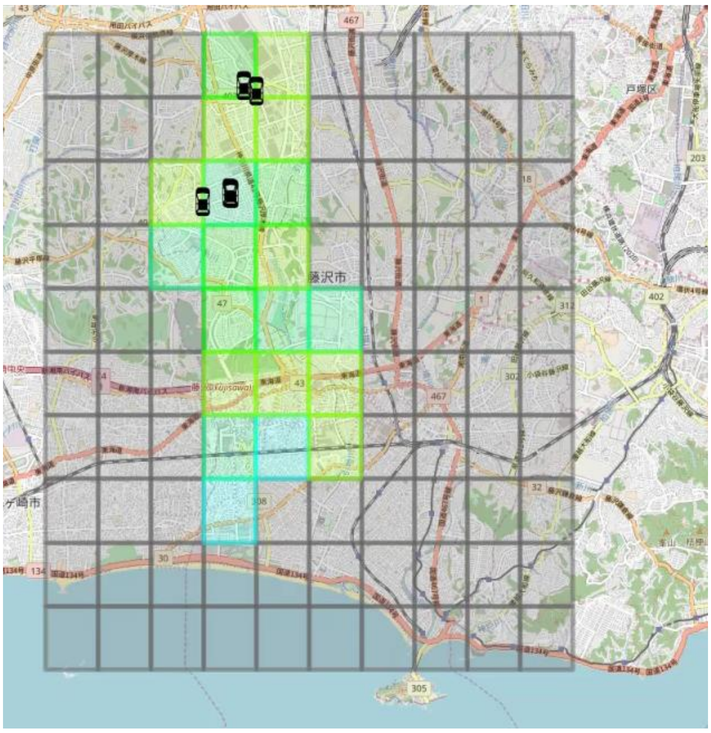


FIGURE 9 OPPORTUNISTIC SENSING IN FUJISAWA CITY

The second application builds a notification application that informs the user that the garbage truck is approaching, by actuating a lamp. Figure 10 illustrates the application.

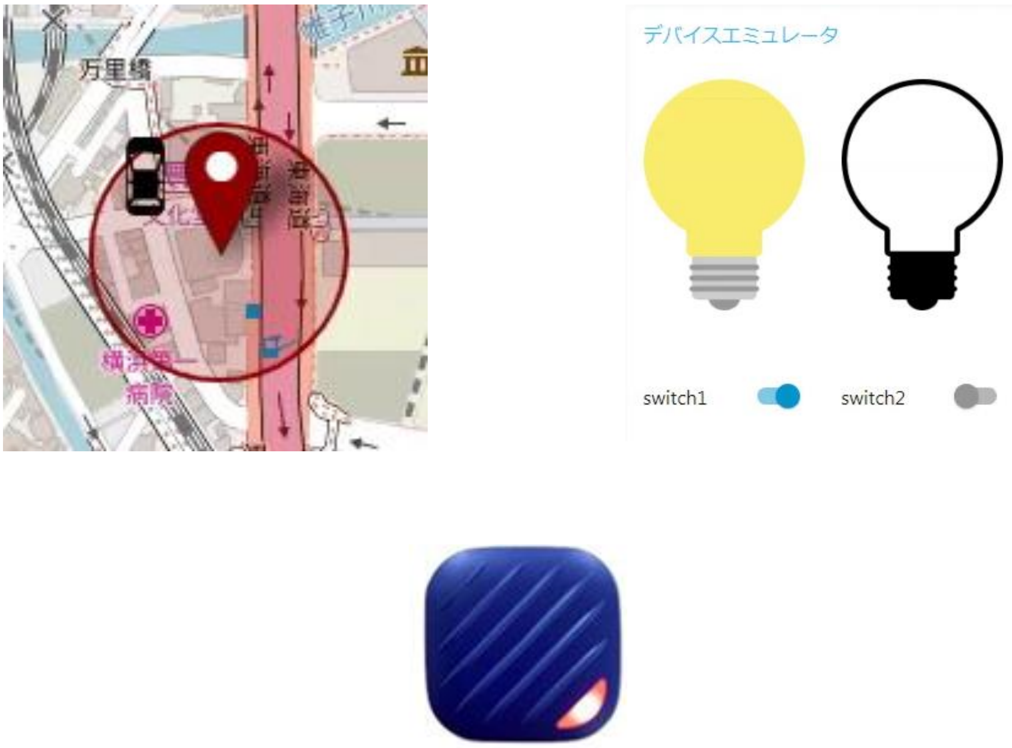


FIGURE 10 I NEED HURRY APP

D-NR has been used to rapidly build the applications. The objective of these application examples is to show that BigClouT platform provides easy ways of collecting data and building applications.

2.3.2.2 SoxFire – BigClouT data lake integration (Keio-LANC)

SoXFire is getting data from Fujisawa trials (garbage truck sensing, lokemon, human flow monitoring, road conditions monitoring, etc.) and transferring it to the BigClouT data lake, either directly or via other platforms such as sensiNact and D-NR. There are already GBs of data of different nature available such as camera images, sensor readings, etc. Data can be found at the bigclout data lake: http://bigclout.lancaster.ac.uk/dataset?tags=fujisawa

After analysis, for instance by KNOWAGE, the analysed data are also sent back to the data lake.

Below a screenshot (Figure 11) from the analysis of data from the garbage trucks by KNOWAGE. A video describing the demo is available at the YouTube channel of BigClouT: https://www.youtube.com/watch?v=wM-6RiJ_6zo

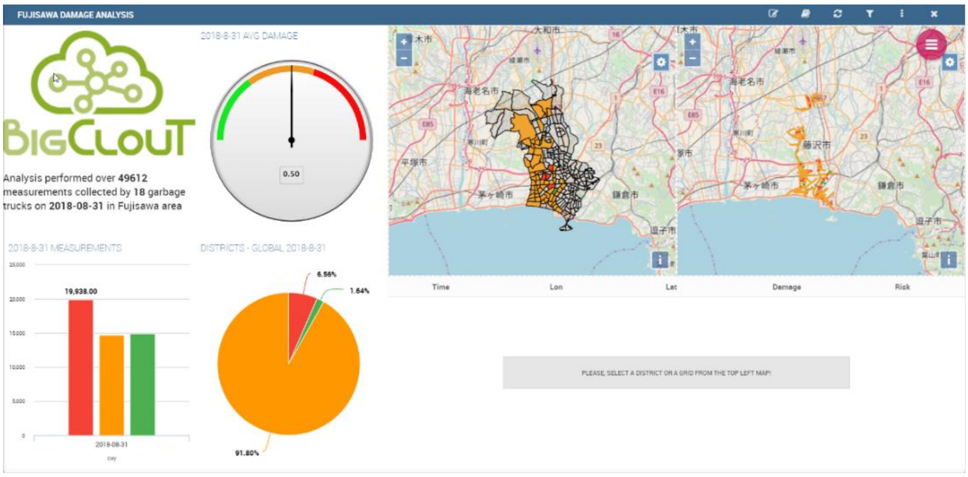


FIGURE 11 FUJISAWA ROAD CONDITIONS ANALYSIS BY SOXFIRE, D-NR AND KNOWAGE

2.3.3 D-NR integrations (LANC)

The D-NR visual programming system consists of two core components, D-NR studio that allows developers to build smart city applications using the BigClouT services, and the D-NR platform that takes smart city applications and distributes them to edge devices in the smart city. Applications are built using pre-defined processing nodes that are combined into application flows.

The Figure 12 below illustrates the use of these extensions allowing a developer, in this case, to specify that a node should run in the 'same city' and that other nodes should be constrained to run on 'server' devices.

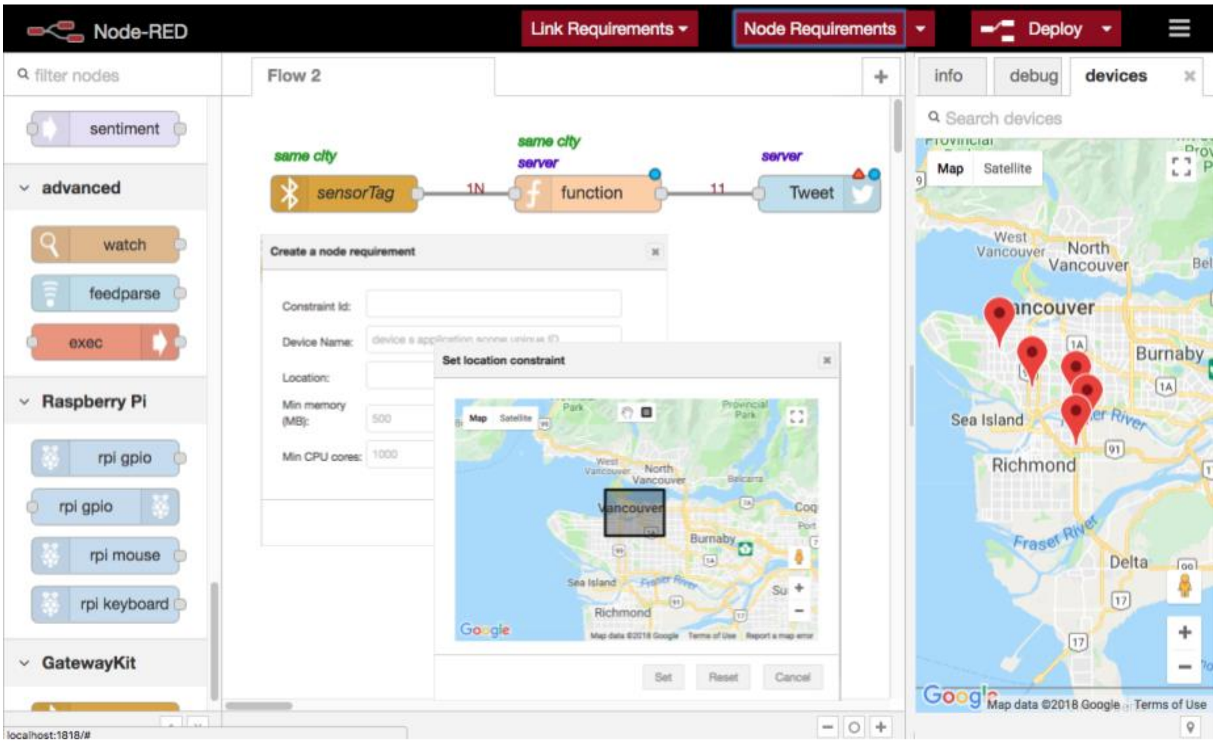


FIGURE 12 CONTROLLING THE PLACEMENT OF CODE ON EDGE DEVICES USING THE CONSTRAINTS.

A number of the BigClouT smart city components and services have been integrated into the D-NR system. These are either integrated using existing D-NR nodes are using newly developed nodes that have been created specifically for the project. In the sections below we discuss the various integration points.

2.3.3.1 D-NR – SOXFire integration (LANC-KEIO)

The SOXFire communications infrastructure has been integrated directly into D-NR using a newly developed SOXFire node. This node employs an XMPP library to talk directly to the SOXFire server allowing smart city application and service developers direct access to pub/sub data managed by SOXFire.

As shown in Figure 13, the Sox-In and Sox-Out nodes can be used to directly access data in SoxFire, e.g., the TruckLocation in the flow below, or to send data to SoxFire after it has been processed in the application flow – again as shown in Figure 13.

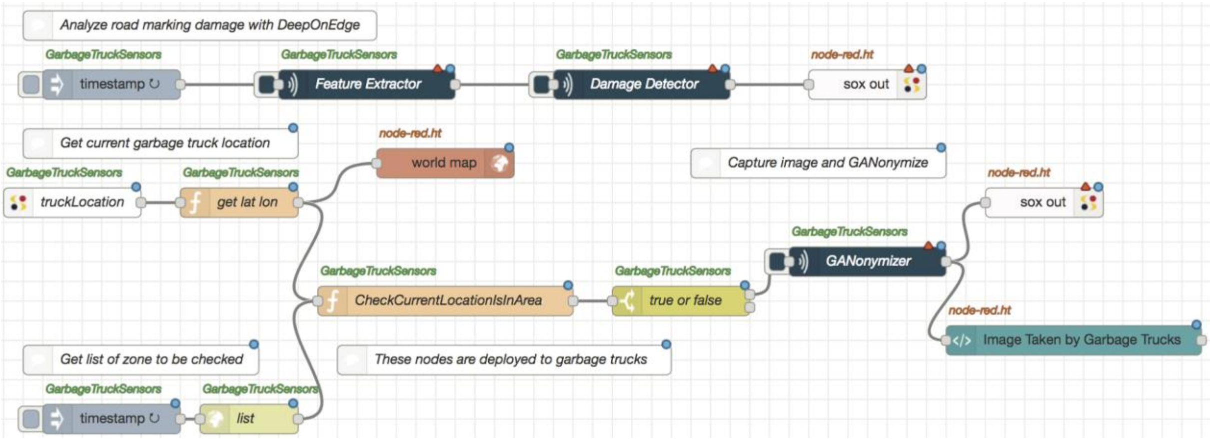


FIGURE 13 ACCESSING AND USING SOXFIRE VIA THE NEWLY DEVELOPED SOX-IN AND SOX-OUT NODES.

These extensions nodes are open source and have been contributed to the base Node-RED community and are available here: https://flows.nodered.org/node/node-red-contrib-sox

2.3.3.2 D-NR - BigClouT datalake integrations (LANC)

The BigClouT Data Lake is tightly integrated into the D-NR system by the development of a new node that talks directly to the Data Lake real-time plugin. This allows BigClouT developers to directly interface with the real-time streams that are being pushed to the Data Lake from a number of Smart City trials.

These nodes allow direct push of new data to a specified Data Lake resource and querying the Data Lake real-time sources for a specific data item using a direct API or a SQL like query.

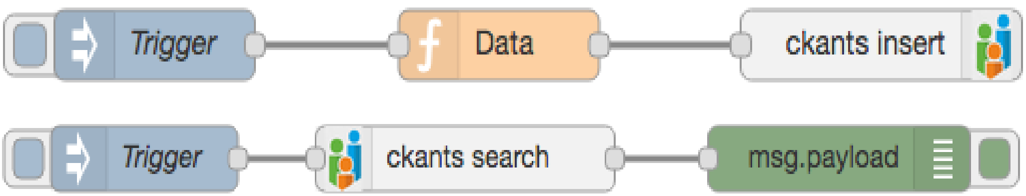


FIGURE 14 ACCESSING AND USING BIGCLOUT DATALAKE VIA THE NEWLY DEVELOPED CKAN TIME SERIES INSERT AND SEARCH NODES

All of the BigClouT trials use the Data Lake in some fashion and those that use D-NR studio make use of these nodes for direct access. Since the Data Lake also has a traditional REST API, it is also possible to directly access Data Lake resources using the built in HTTP API node. Although less integrated than the direct access through the insert and search nodes, the HTTP API nodes provides a generic interface for all non-real-time resources managed by the Data lake.

These extensions nodes are open source and have been contributed to the base Node-RED community and are available here: https://flows.nodered.org/node/node-red-contrib-ckants

2.3.3.3 D-NR – DNR Studio integrations (LANC)

As described above and shown in Figure 11, the D-NR system consists of the visual programming tool D-NR studio and the underlying D-NR platform. D-NR platform provides an automatic edge placement and processing system that takes BigClouT smart city applications and automatically distributes them to edge devices in the smart city network. The complexity of this process is hidden from the developers and simply requires that they express constraints via the visual D-NR studio. These constraints then allow the underlying D-NR platform to decide on placement to meet the constraints – essentially executing a distributed constraint satisfaction algorithm.

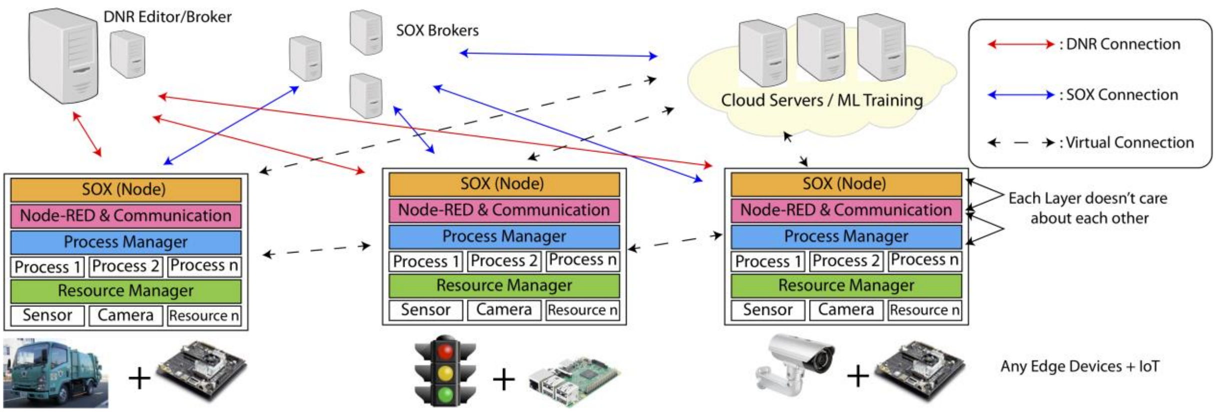


FIGURE 15 MANAGING A VARIETY OF EDGE NODES IN A D-NR BASED SMART CITY DEPLOYMENT

2.3.3.4 D-NR - DeeponEdge integrations (LANC-Keio)

The DeepOnEdge processing node developed by the Keio team is used extensively in the Fujisawa trial. It provides an AI based capability to analyse real time images that are captured by the Fujisawa garbage trucks. As shown in Figure 16, this allows smart city application developers to easily use and control DeepOnEdge components in their application flows and thus exploit advanced smart city analysis tools.

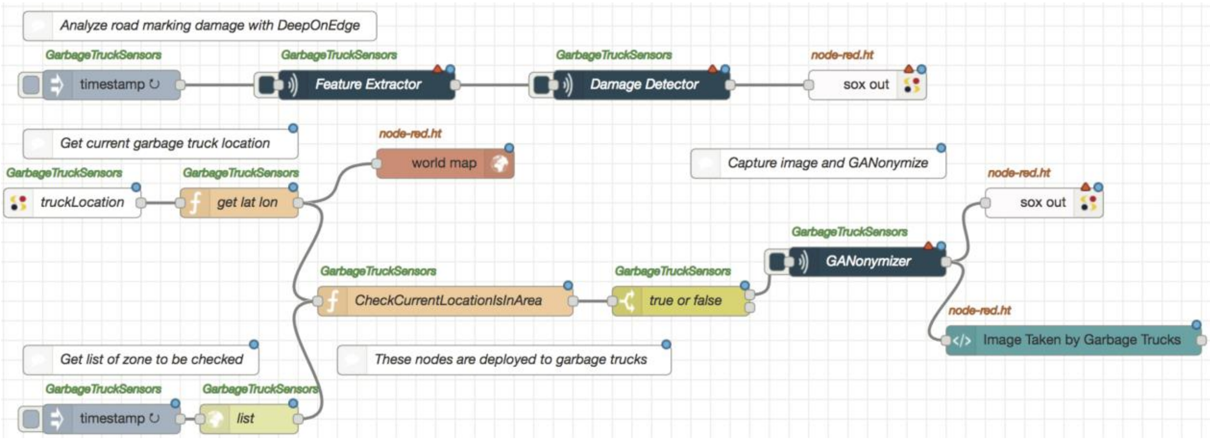


FIGURE 16 ACESSING DEEPONEDGE COMPONENTS IN A D-NR STUDIO FLOW

In this case, components of the DeepOnEdge service, Feature extraction, Damage Detection and GANonymizer are combined into a flow (that also uses the SOXFire In/OUT nodes) to capture images, perform processing and write results.

2.3.4 StreamingCube Integrations (TSU)

2.3.4.1 StreamingCube – D-NR integrations (TSU-LANC)

StreamingCube is integrated with D-NR so that users can use StreamingCube to process data streams in real-time and to perform OLAP-style multi-dimensional analysis over data streams. The actual integration is done by the developments on both StremingCube and D-NR sides. On StreamingCube side, to make D-NR (and other systems as well) possible to send JSON streams to StreamingCube for subsequent stream processing/analysis, we have developed an intermediate Web API server that takes as input HTTP request containing a piece of JSON data and pass it to StreamingCube. Thus, clients can easily send JSON data to StreamingCube by simple HTTP request. Besides, on D-NR side, a new node that communicates with the intermediate Web API server has been developed. It allows D-NR users to process JSON streams using StreamingCube in the D-NR environment.

2.3.4.2 StreamingCube – sensiNact - SOXFIRE integration (TSU-CEA)

Sensors attached to garbage collecting cars in Fujisawa city collect real-time environmental data with respect to different areas in the city. Those environmental data are collected by SOXFIRE and made accessible to StreamingCube by using sensiNact. In this demonstration, the following environmental data are used:

- **Geolocation of the carsensors**: It can be interpreted into city, block, machi, and chome. Note that, machi and chome are terms that are used to name the address in Japan. The sizes of city, block, machi, and chomi are from bigger to smaller following this order.
- **PM2.5 value**: The value of PM2.5 at all places where the garbage trucks go through.

The table in Figure 17 (top left) shows the average PM2.5 values in real time of the selected area (e.g., chome) w.r.t. the time (e.g., seconds). The chart on the top right visualizes the results based on both area and time dimensions. It shows the percentage of the combined dimensions of all valid environmental data that has been received so far. To meet different needs of the users, StreamingCube provides a wide range of charts for interpreting the results. All available charts (Bar, Column, Area, Spline, Pie, and Doughnut) are shown in Figure 17. The bar chart on the bottom left shows the average PM2.5 w.r.t. the time dimension regardless of the areas.

The interactive map (bottom right in Figure 17) visualizes the results in an interactive map.

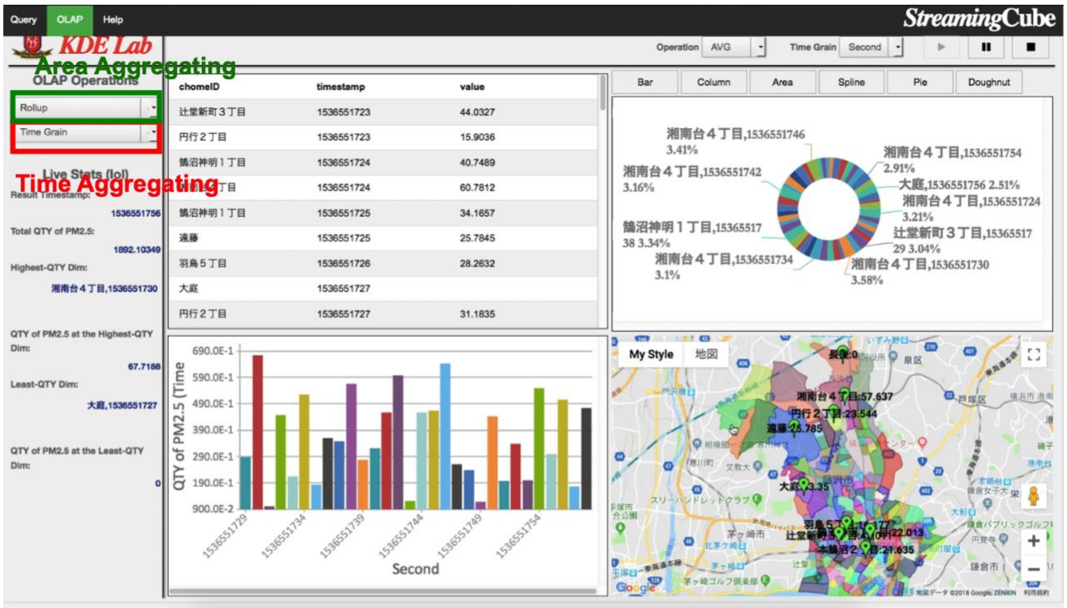


FIGURE 17 OUTPUT SHOWING THE AMOUNT OF PM2.5 WITH RESPECT TO DIFFERENT AREAS

More details about the integration work can be found in the Deliverable 2.6 [10].

2.3.5 KNOWAGE integrations (ENG)

KNOWAGE is placed at the BigData Analysis and Visualization layers of the BigClouT's platform. Its main role is extracting valuable information out of data coming from data sources, which can provide useful insights to the cities. This section of the document describes the integrations between KNOWAGE and others BigClouT's components: BigClouT Data Lake, Recommendation Service and Edge Storage.

2.3.5.1 KNOWAGE – BigClouT Data Lake (ENG)

KNOWAGE provides a specific CKAN connector to retrieve BigClouT Data Lake's data as raw CSV or XLS file. Figure 18 depicts an example of CKAN dataset configuration in KNOWAGE. In order to configure such datasets, the user has to specify the ID of the resource and URL of the file provided by CKAN. Moreover, in the case of a CSV file, the user has to specify also the delimiter character, the quote character, the encoding and the date format of the file.

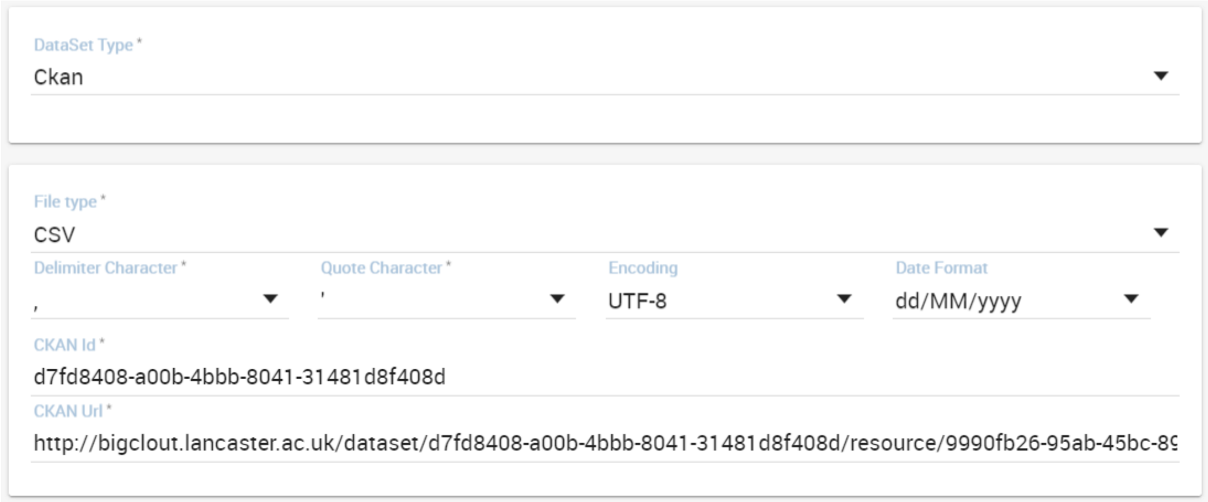


FIGURE 18 KNOWAGE - CKAN DATASET

In example depicted in Figure 18, the configured BigCLouT Data lake's dataset refers to the so-called 'ckan_default_DB' resource[4] of the 'Data Lake-stats[5]'. This resource, together with the 'datastore_default_DB[6]' and the 'file-store-size[7]' resources are analysed and the obtained results are used to build the 'Data Lake' Statistics' dashboard (Figure 19) whose link is the following:

https://knowage.opsi-lab.it/knowage/public/servlet/AdapterHTTP?ACTION_NAME=EXECUTE_DOCUMENT_ACTION&OBJECT_LABEL=DataLake_Aggregated&TOOLBAR_VISIBLE=false&ORGANIZATION=BigClouT&NEW_SESSION=TRUE&SBI_LANGUAGE=en&SBI_COUNTRY=US

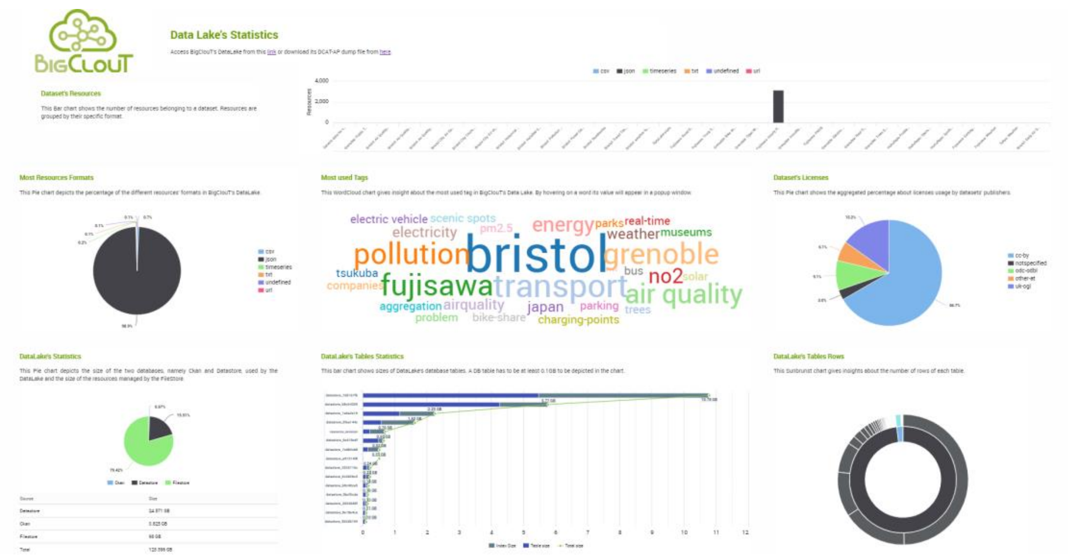


FIGURE 19 KNOWAGE - DATA LAKE'S STATISTICS DASHBOARD

4 http://bigclout.lancaster.ac.uk/dataset/datalake-stats/resource/9990fb26-95ab-45bc-8980-7f01fd7e0d42
5 http://bigclout.lancaster.ac.uk/dataset/d7fd8408-a00b-4bbb-8041-31481d8f408d
6 http://bigclout.lancaster.ac.uk/dataset/datalake-stats/resource/c0f440c8-5e66-473a-a1d9-d8aada7be8d7
7 http://bigclout.lancaster.ac.uk/dataset/datalake-stats/resource/3048bdc5-6e9e-4b42-b83a-cde6d1fd722a

Thanks to the Data Lake's DataStore APIs, KNOWAGE it is also able to retrieve data from the BigClouT Data Lake by configuring a REST dataset in KNOWAGE.

Thanks to the KNOWAGE – BigClouT Data Lake integration, all data that are in the lake can be analysed and visualised by the KNOWAGE.

For instance, KNOWAGE has been used for the Fujisawa trial to analyse the pollution data and provide useful analysis and dashboards for the city. Through the analysis of environmental data related to Fujisawa's area and available on the BigClouT Data Lake[8] a dashboard has been built leveraging KNOWAGE's functionalities.

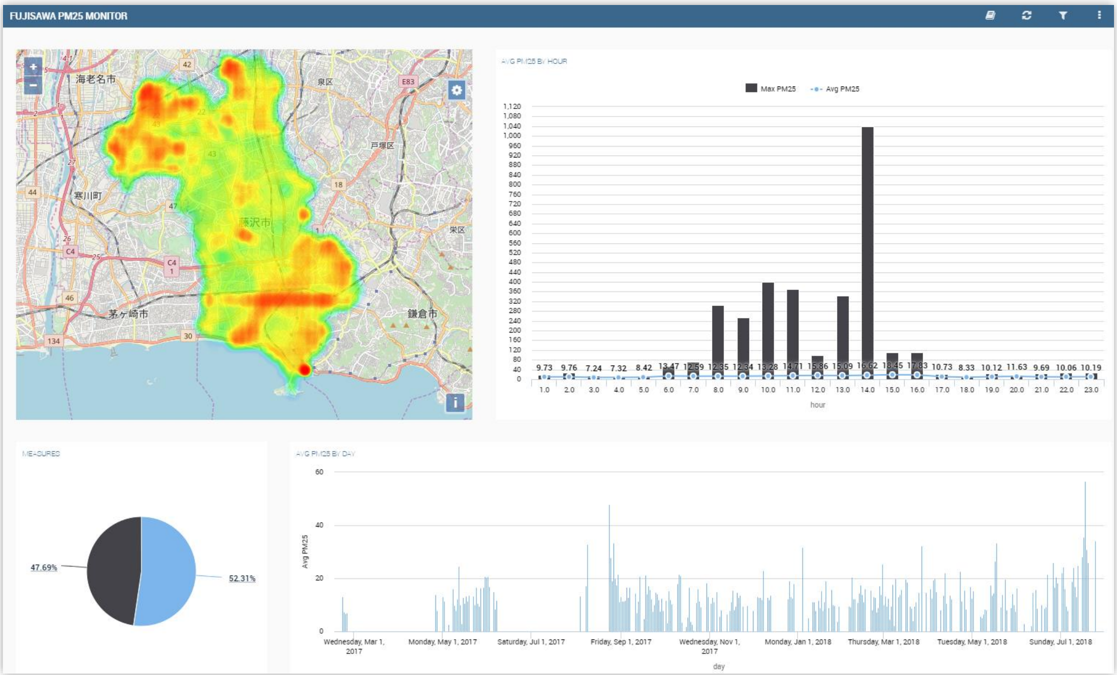


FIGURE 20: KNOWAGE - FUJISAWA PM25 MONITOR DASHBOARD

Figure 20 depicts the resulting dashboard which can be accessed through the following link: https://knowage.opsi-lab.it/knowage/public/servlet/AdapterHTTP?ACTION_NAME=EXECUTE_DOCUMENT_ACTION&OBJECT_LABEL=FujisawaPm25&TOOLBAR_VISIBLE=true&ORGANIZATION=BigClouT&NEW_SESSION=true&PARAMETERS=year=&year_field_visible_description=&month=&month_field_visible_description=

The rationale behind this dashboard is to give insights about PM2.5's trend during the provided time interval. Each widget shows the average PM2.5 values obtained by performing different aggregation methods to the punctual measurements given as input from the source files. Specifically:

- The heatmap widget (Figure 21) gives the detail of the average PM2.5 values in the city area.
- The top-right bar chart widget (Figure 22) shows the average and maximum PM2.5 values aggregating the data by the specific hour of the day.

[8] http://bigclout.lancaster.ac.uk/dataset/fujisawa-hourly-pm2-5-density-by-mesh

- The bottom-right bar char widget (Figure 23) depicts the average PM2.5 values aggregating the data taken in a single day.
- The pie chart widget (Figure 24) illustrates the number of measures gathered by sensors per year. Furthermore, by clicking on a specific year, the user will see the month detail of the measures.

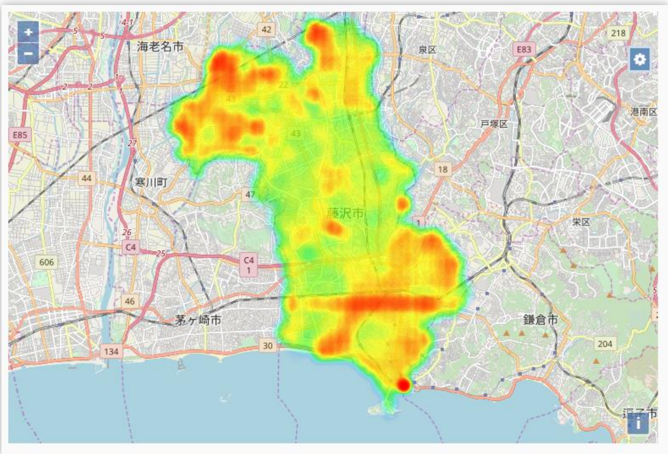


FIGURE 21: KNOWAGE - FUJISAWA PM25 HEATMAP

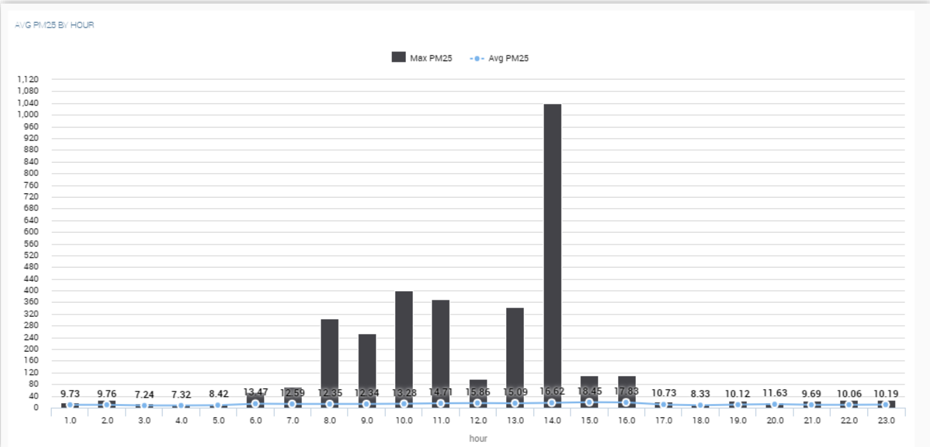


FIGURE 22: KNOWAGE - FUJISAWA AVERAGE AND MAXIMUM PM25 PER HOUR

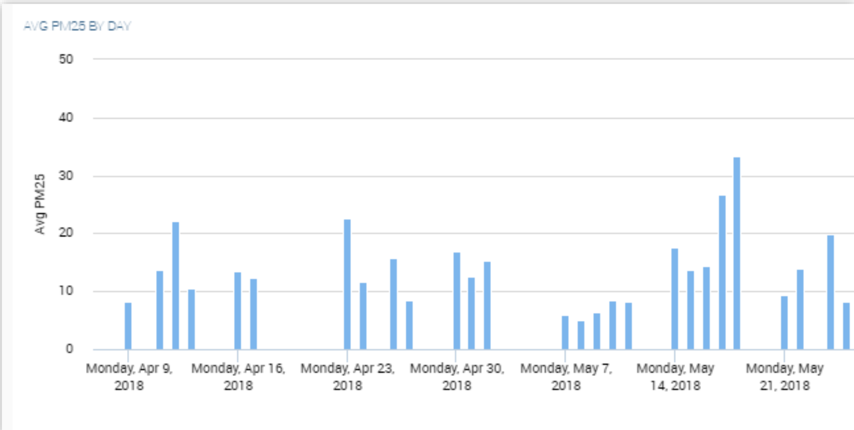


FIGURE 23: KNOWAGE - FUJISAWA AVERAGE PM25 PER DAY

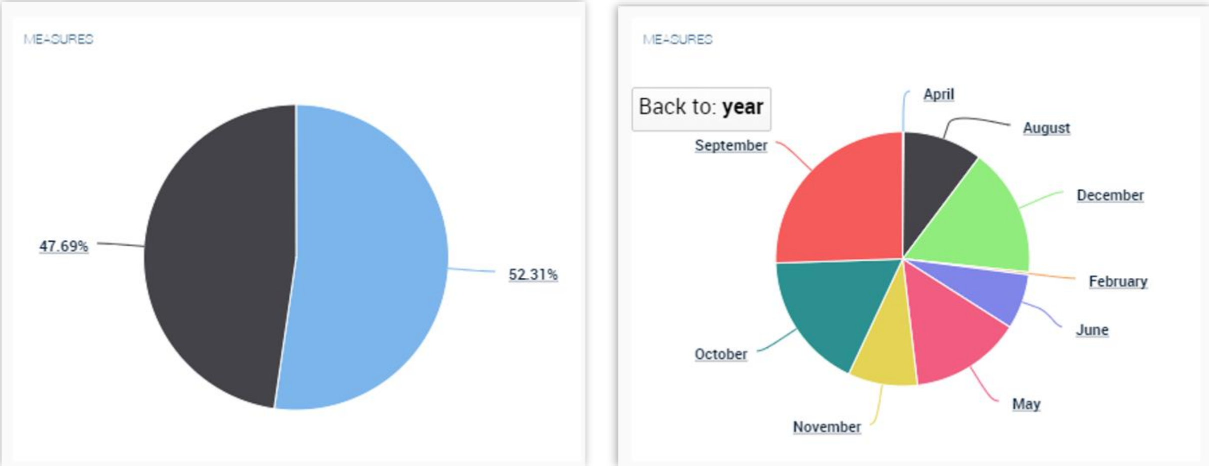


FIGURE 24: KNOWAGE - FUJISAWA PM25 PIE CHART DETAIL

In this dashboard the user can perform the following interactions:

- By clicking in a specific day in the bottom-right bar chart, the widgets will update showing only the data taken that day.
- By selecting a point in the map, a popup reporting its detail will appear and the other widgets will update accordingly showing the details of the selected area, as depicted in Figure 25.

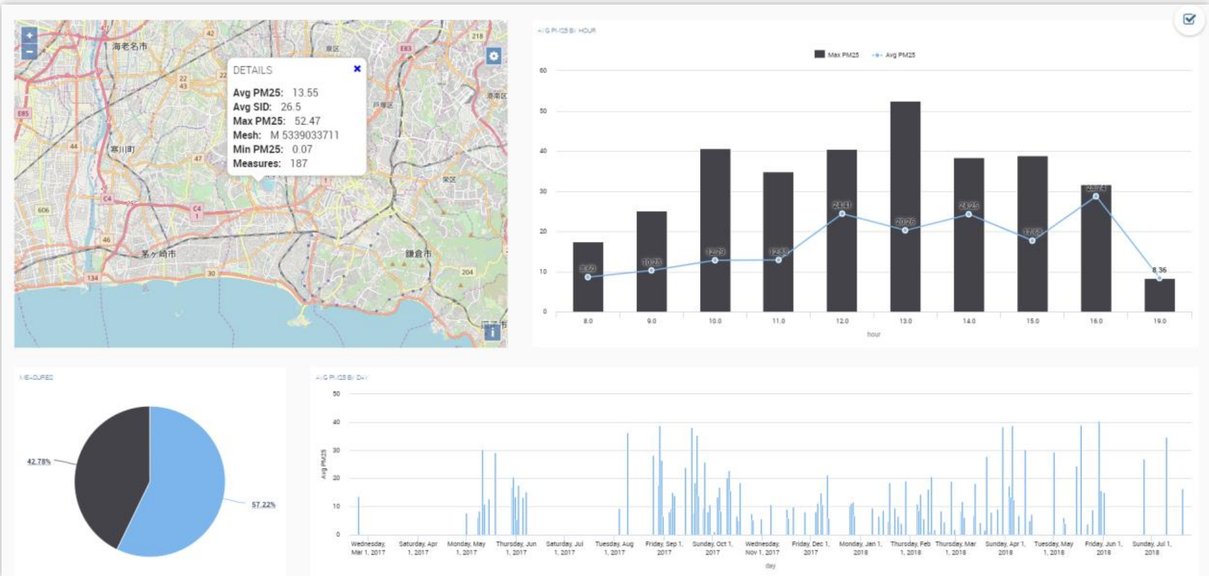


FIGURE 25: KNOWAGE - FUJISAWA PM25 MAP WIDGET INTERACTION

Moreover, a year and a month filters are provided to let the user compare specific time period, as illustrated in Figure 26. By selecting the filters values and clicking on the 'Execute' button, the dashboard will show the information of the selected year(s) and month(s).

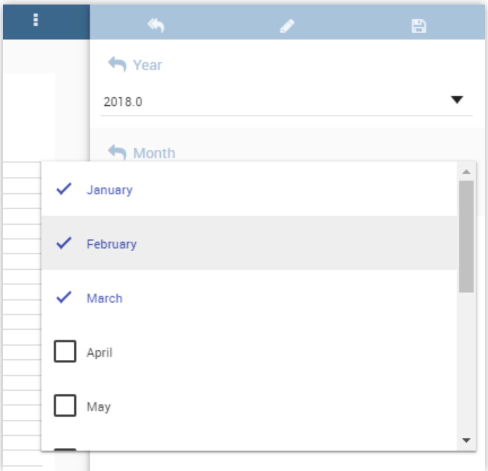


FIGURE 26: KNOWAGE - FUJISAWA PM25 FILTERS

Similarly, the data from the Tsukuba city trial available in the Data Lake allowed visualising useful information such as geographic distribution of posts, categories, posts per users, etc. as illustrated in Figure 27.

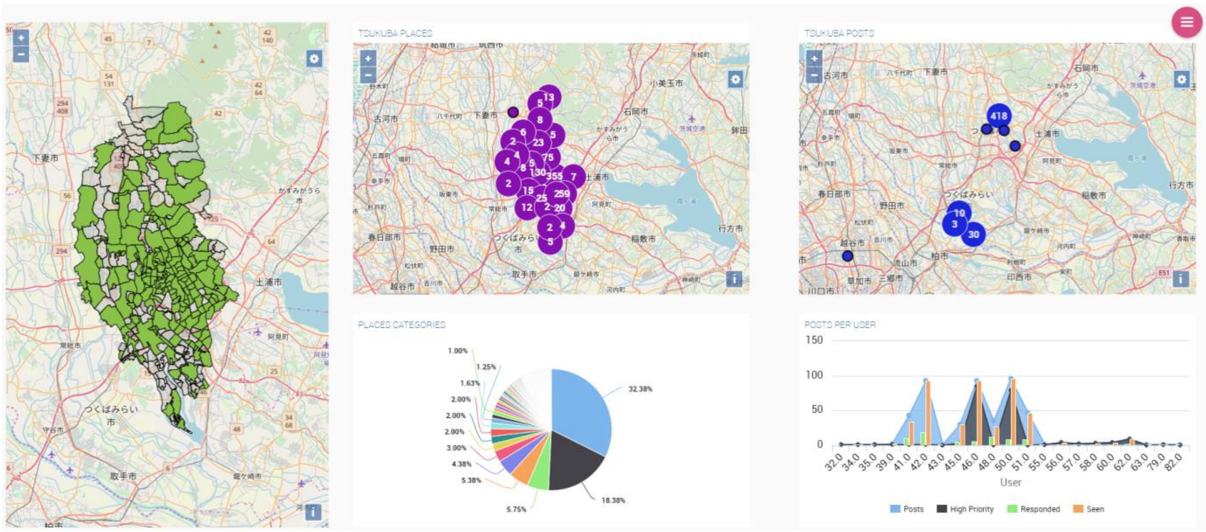


FIGURE 27. DATA ANALYSIS OF HUKUREPO DATA BY KNOWAGE

2.3.5.2 KNOWAGE – Recommendation Service (ENG - ICCS)

KNOWAGE integration with the Recommendation Service is performed taking advantage of the RESTful APIs exposed by the latter component. In KNOWAGE, a REST dataset is used to access results provided by the Recommendation Service and to perform analysis and create visualisation. Thanks to the parameters management section of the REST dataset, it is also possible to query the Recommendation Service's APIs to retrieve data fulfilling the specific parameter, if any exists.

FIGURE 28 KNOWAGE - RECOMMENDATION SERVICE DEMO DASHBOARD

Figure 28 depicts a hypothetic dashboard, built from demo data provided by the Recommendation Service through its REST APIs, to check about energy consumption in houses. Thanks to the filters in the right panel, it is possible to execute a real time query to the Recommendation Service. The results are then used to populate the chart widgets following their specific aggregation and visualization settings. It is important to underline that each filter manages the value of a configured query parameter to be provided to the Recommendation Service.

2.3.5.3 KNOWAGE – Edge Storage (ENG)

As in the case of KNOWAGE and Recommendation Service integration, the integration between KNOWAGE and Edge Storage is performed taking advantage of KNOWAGE's REST connector. Data to be analysed and visualized are gathered from Edge Storage thanks to its CDMI 1.1.1 compliant RESTful APIs.

Within KNOWAGE, Object and Queues CDMI REST APIs have been tested to gather data from Edge Storage. Since CDMI response returns data values in its metadata field, KNOWAGE's REST connector gives the opportunity to define the so called JSON Path Attributes to extract nested fields to be used in analysis or visualization. For instance, the snippet reported in Table 2 provides an example of CDMI's Object response

TABLE 2 CDMI'S OBJECT RESPONSE SNIPPET

```
{
      "parentURI": "KnowageBigClouT",
      "objectName": "ObjectTest",
      "mimetype": "application/json",
      "value": "50"
      "objectID": "0000000B0018ADE738303334666434312D356230392D3438",
      "metadata": {
            "latitude": "40",
            "last_update": "20190729102053",
            "longitude": "30"
      }
}
```

To access latitude, longitude and last_update fields for analysis and visualizations, the KNOWAGE's REST dataset has to be configured with the appropriate JSON Path Attributes; an example is depicted in Figure 29.

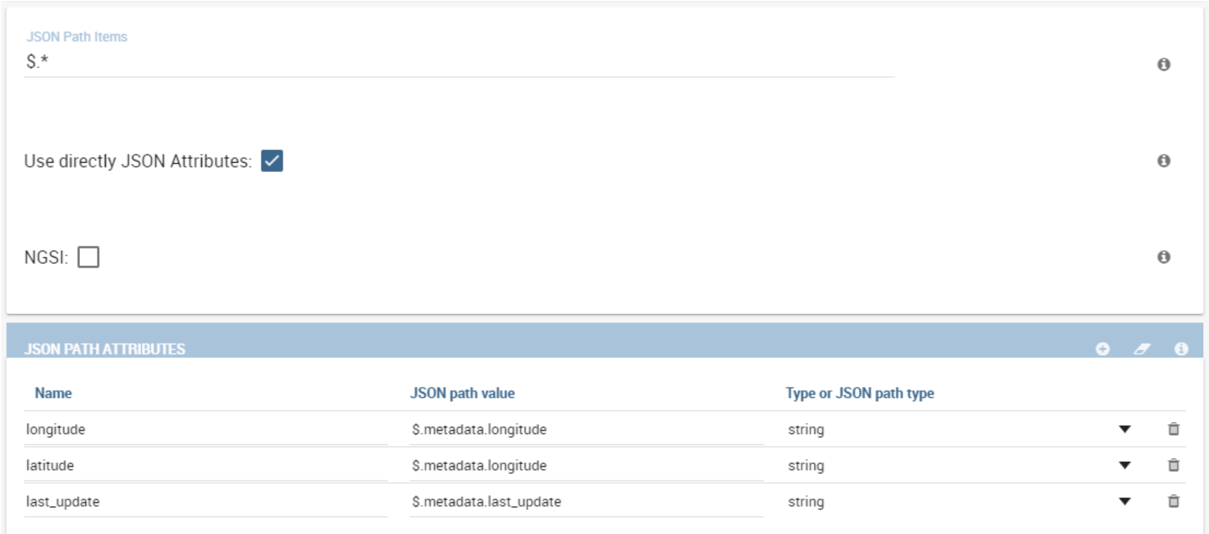


FIGURE 29 KNOWAGE - JSON PATH ATTRIBUTES

It is important to underline that thanks to the redirect mechanism provided by the Edge Storage, following the Fog Paradigm, KNOWAGE can gather data from several nodes using the same endpoint, since the Edge Storage manages redirections to the other nodes (master or edge nodes) if the requested data are not in the node referred by the endpoint.

2.3.6 Recommendation Service – BigClouT data lake integration (ICCS)

The BigClouT Recommendation Service APIs are using three different datasets, all hosted in the BigClouT CKAN Data Lake.

The Bristol GreenPath recommendation use case is reading forecast data about the predicted pollutants in each location of the city. This data extraction is executed by a Python v3 script, which downloads the csv dump of the dataset, transforms it into a JSON format correctly formatted for quick MongoDB import and then imports the JSON data into an NTUA hosted MongoDB database. Then a Node-RED flow reads the data from the MongoDB and using a Python API calculates a score for various possible paths through the city, recommending the less polluted route.

The Tsukuba GreenPath recommendation use case is functioning the same way as the Bristol one, with the exception of the initial data format. The Tsukuba pollutants dataset is comprised of historical data and no forecasts, having a different grid geometry. So, as a pre-processing step, we are using a Python v3 script to request the dataset from the CKAN API, transforming it into one that fits the Bristol dataset, and then import it into the same MongoDB database. The Python recommendation API then calculates a slightly different model, taking into account the difference between the historical and forecasted datapoints and the new grid geometry.

The Fujisawa Point of Interest (PoI) recommendation use case is reading data from the CKAN, creating a new collection in our MongoDB server which contains all the data about the points of interest. A Node-RED flow is receiving user requests, containing the point of origin and the destination of the route to be calculated as well as the types of PoIs that the user is interested in. Then it calculates a new route that passes close to at least one PoI of each type specified, with as little disturbance to the original route as possible. Figure 30 is a screenshot of the recommendation service showing PoI in Fujisawa.

FIGURE 30: THE REECOMMENDATION SERVICE SHOWING POINTS OF INTEREST IN FUJISAWA.

2.3.7 Integration Readiness Level Analysis

The below Table 3 and Table 4 summarize respectively the initial and final integration readiness levels (IRL) of each integration work. IRL concept is similar to the TRL, focusing on the integration work between technologies (See Table 5 for the description of different IRLs). Out of the TRL and IRL levels, we calculate the overall System Readiness Level (SRL) which is an indicator of the maturity of the solution (See Table Table 6. Please refer to Annex Section 5 and to [17] for details on calculating the SRL.

TABLE 3 INITIAL INTEGRATION READINESS LEVELS

Initial IRL per integration	BigClouT data lake	DeepOnEdge	D-NR platform	D-NR studio	ECA verifier	Edge storage	Knowage	oneM2M	Recommendation service	SensiNact	SensiNact Studio	SOXFire	StreamingCube
BigClouT data lake	9	0	1	0	0	0	1	0	1	1	0	1	1
DeepOnEdge		9	1	0	0	0	0	0	0	0	0	0	0
D-NR platform			9	1	0	0	0	1	0	0	0	1	0
D-NR studio				9	0	0	0	0	0	0	0	0	0
ECA verifier					9	0	0	0	0	0	2	0	0
Edge storage						9	1	0	0	2	0	0	0
Knowage							9	0	1	0	0	0	0
oneM2M								9	0	3	0	1	0
Recommendation service									9	0	0	0	0
SensiNact										9	6	3	1
SensiNact Studio											9	0	0
SOXFire												9	0
StreamingCube													9

TABLE 4 FINAL INTEGRATION READINESS LEVELS

Final IRL per integration	BigClouT data lake	DeepOnEdge	D-NR platform	D-NR studio	ECA verifier	Edge storage	Knowage	oneM2M	Recommendation service	SensiNact	SensiNact Studio	SOXFire	StreamingCube
BigClouT data lake	9	0	6	0	0	0	9	0	5	6	0	5	5
DeepOnEdge		9	5	0	0	0	0	0	0	0	0	0	0
D-NR platform			9	6	0	0	0	5	0	0	0	5	0
D-NR studio				9	0	0	0	0	0	0	0	0	0
ECA verifier					9	0	0	0	0	0	4	0	0
Edge storage						9	3	0	0	3	0	0	0
Knowage							9	0	4	0	0	0	0
oneM2M								9	0	4	0	5	0
Recommendation service									9	0	0	0	0
SensiNact										9	7	5	6
SensiNact Studio											9	0	0
SOXFire												9	0
StreamingCube													9

The following Table 5 provides the description of levels of readiness in case of integrations.

TABLE 5 INTEGRATION READINESS LEVEL DESCRIPTIONS

IRL Level	Integration Readiness Level Description
9	Integration is Mission Proven through successful mission operations.
8	Actual integration completed and Mission Qualified through test and demonstration in the system environment.
7	The integration of technologies has been verified and validated with sufficient detail to be actionable.
6	The integrating technologies can accept, translate, and structure information for its intended application.
5	There is sufficient control between technologies necessary to establish, manage, and terminate the integration.
4	There is sufficient detail in the quality and assurance of the integration between technologies.
3	There is Compatibility (i.e., common language) between technologies to orderly and efficiently integrate and interact.
2	There is some level of specificity to characterize the interaction (i.e. ability to influence) between technologies through their interface.
1	An interface (i.e. physical connection) between technologies has been identified with sufficient detail to allow characterization of the relationship.

TABLE 6 SYSTEM READINESS LEVEL DESCRIPTIONS

SRL	Name	Definition
5	*Operations & Support*	Execute a support program that meets operational support performance requirements and sustains the system in the most cost-effective manor over its total life cycle.
4	*Production & Development*	Achieve operational capability that satisfies mission needs.
3	*System Development & Demonstration*	Develop a system or increment of capability; reduce integration and manufacturing risk; ensure operational supportability; reduce logistics footprint; implement human systems integration; design for producibility; ensure affordability and protection of critical program information; and demonstrate system integration, interoperability, safety, and utility.
2	*Technology Development*	Reduce technology risks and determine appropriate set of technologies to integrate into a full system.
1	*Concept Refinement*	Refine initial concept. Develop system/technology development strategy

3. CONCLUSIONS, FINAL BIGCLOUT PLATFORM AND TRIAL DEMONSTRATIONS

13 components have been developed/adapted and about 20 integration links have been established between the components to fulfil the requirements from the BigClouT trials. The Figure 31 illustrates the complete platform with the components mapped to the functional layers.

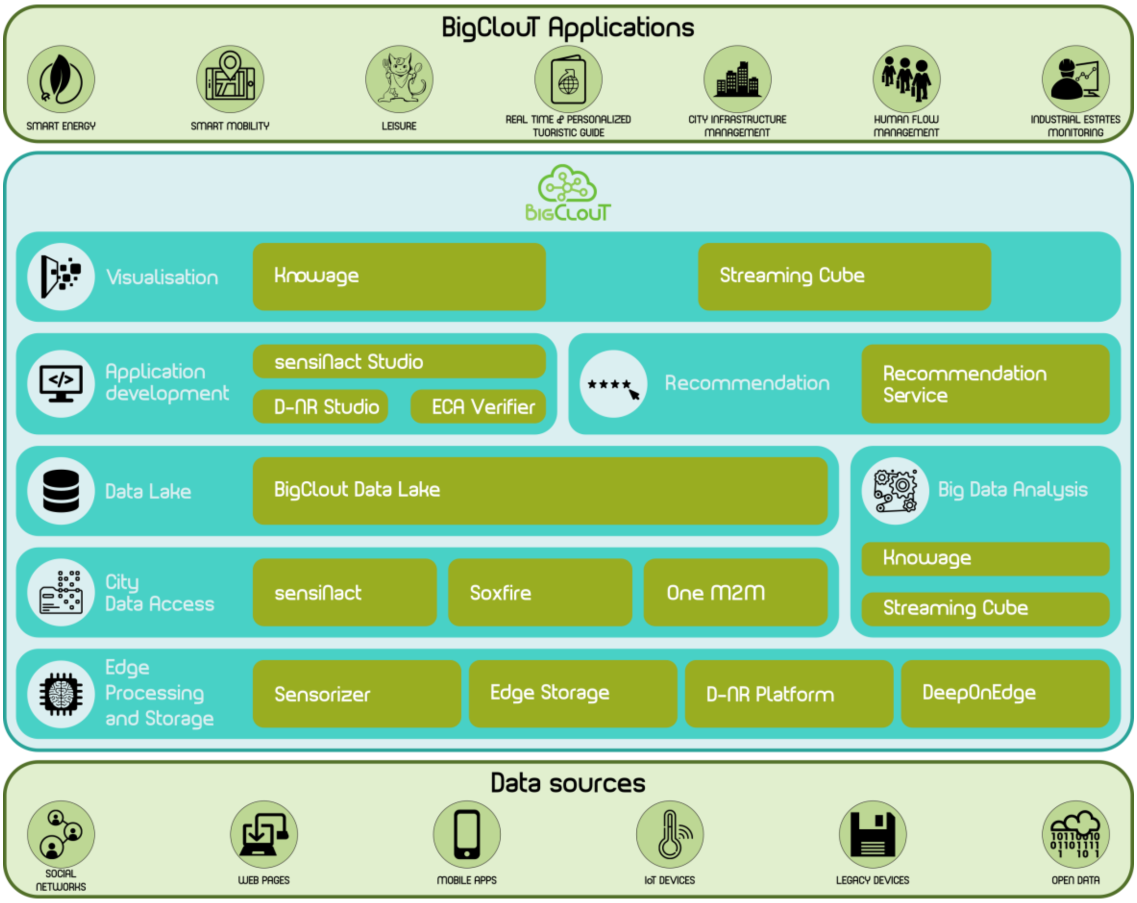
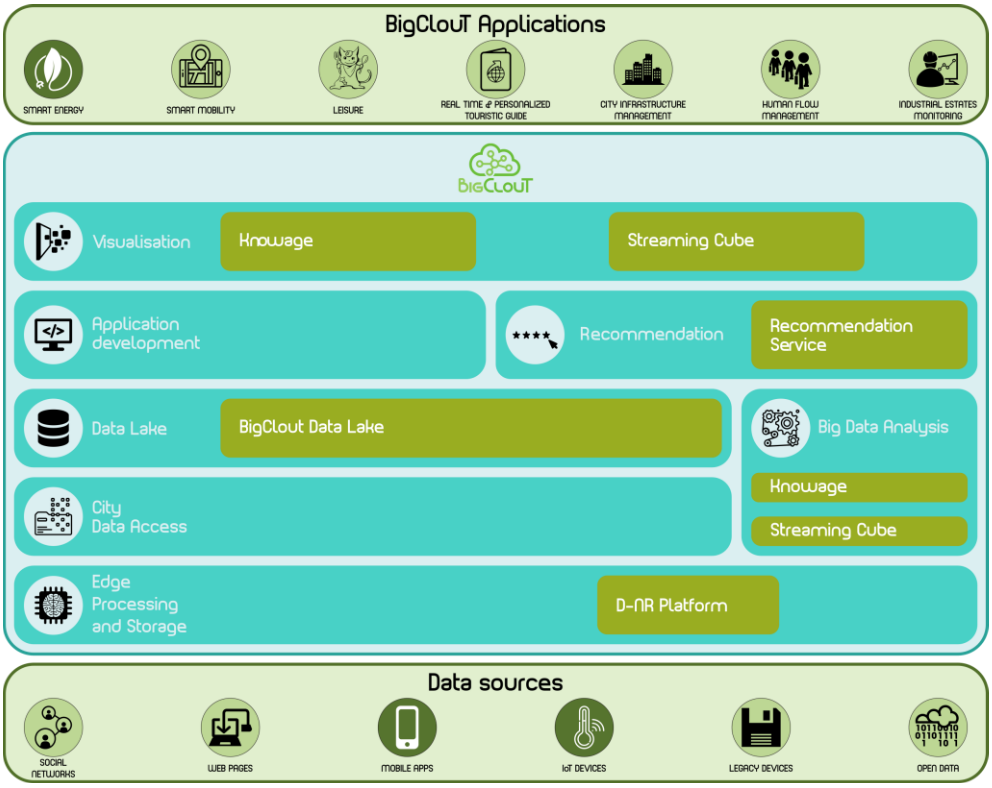


FIGURE 31 BIGCLOUT PLATFORM COMPOSED OF INDIVIDUAL COMPONENTS DEVELOPED AND ADAPTED IN THE PROJECT

The integrated platform has been used and tested in close to real life environments, which allowed us to increase the maturity level of each component, as well as the integrated results. The initial and final readiness levels of the components have been given in the Section 2.2. Similarly, the readiness levels of integration points have been given in the Section 2.3. Combination of these levels provided us the readiness levels (SRL) of each BigClouT instance for each trial case.

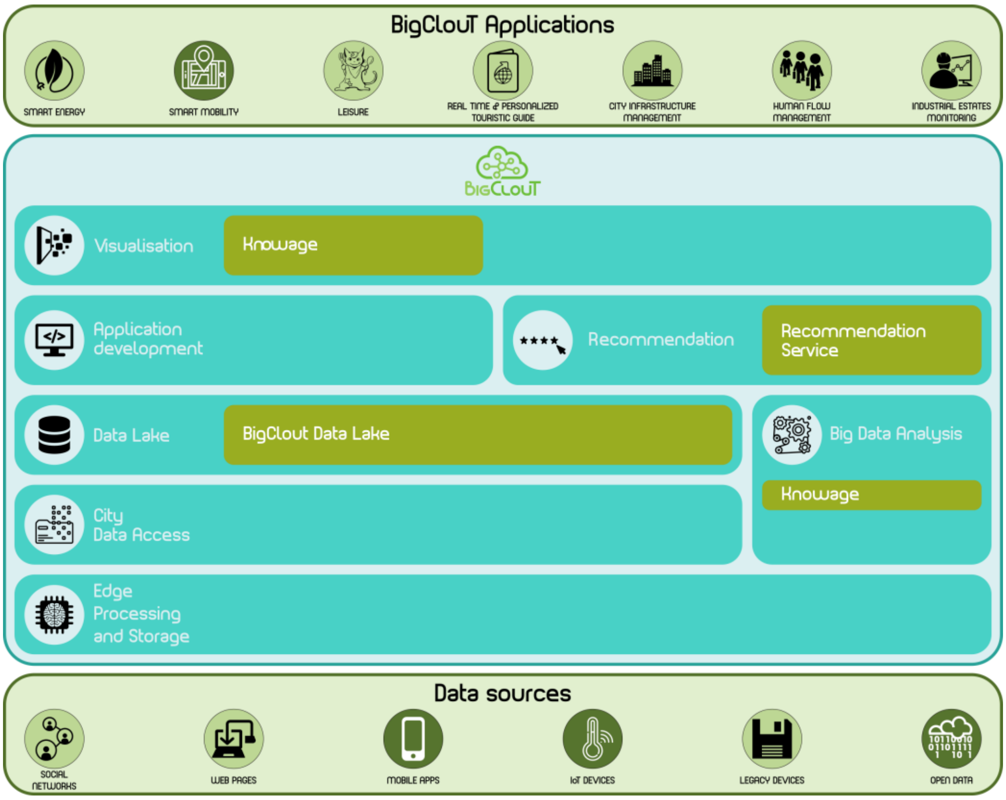
Each trial has used a customised instance of the platform composed of necessary functional components. The following figures show the each instance of BigClouT customised for the given trial, as well as their system readiness level. More details about each trial can be found in the Deliverable 4.5



SRL

	Beginning	End		cSRL Begining	cSRL end
BigClouT data lake	0,26	0,65	V	0,22	0,63
D-NR platform	0,22	0,67	L	1	3
Knowage	0,23	0,69			
Recommendation service	0,16	0,52			

FIGURE 32 SMART ENERGY TRIAL IN BRISTOL - OVERALL PICTURE AND SYSTEM READINESS LEVEL = 3 (SYSTEM DEVELOPMENT AND DEMONSTRATION)



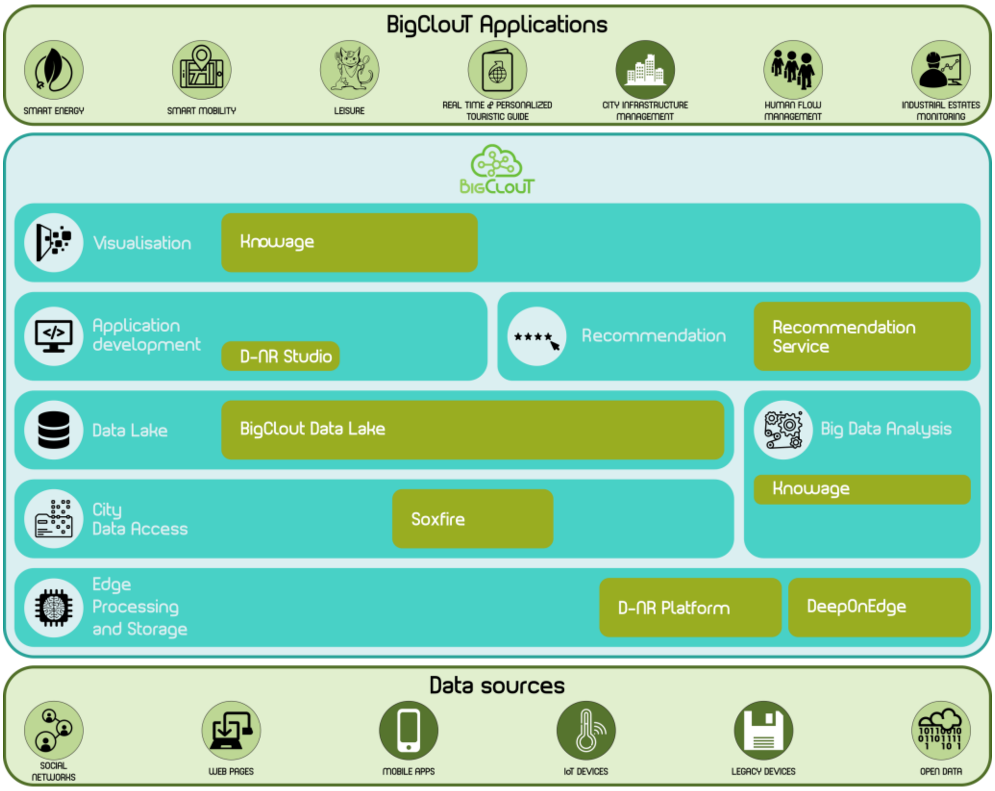
SRL

	Beginning	End
BigClouT data lake	0,33	0,72
Knowage	0,23	0,69
Recommendation service	0,16	0,52

cSRL

	Begining	end
V	0,24	0,64
L	1	3

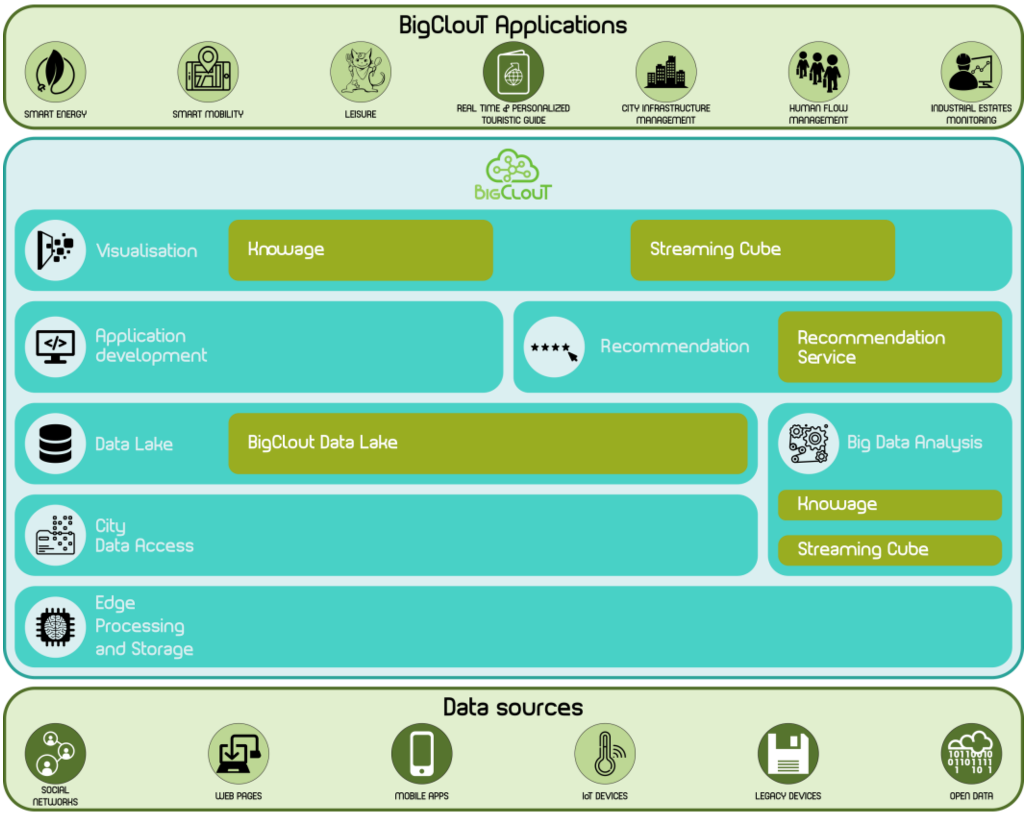
FIGURE 33 SMART MOBILITY TRIAL IN BRISTOL- OVERALL PICTURE AND SYSTEM READINESS LEVEL = 3 (SYSTEM DEVELOPMENT AND DEMONSTRATION)



SRL

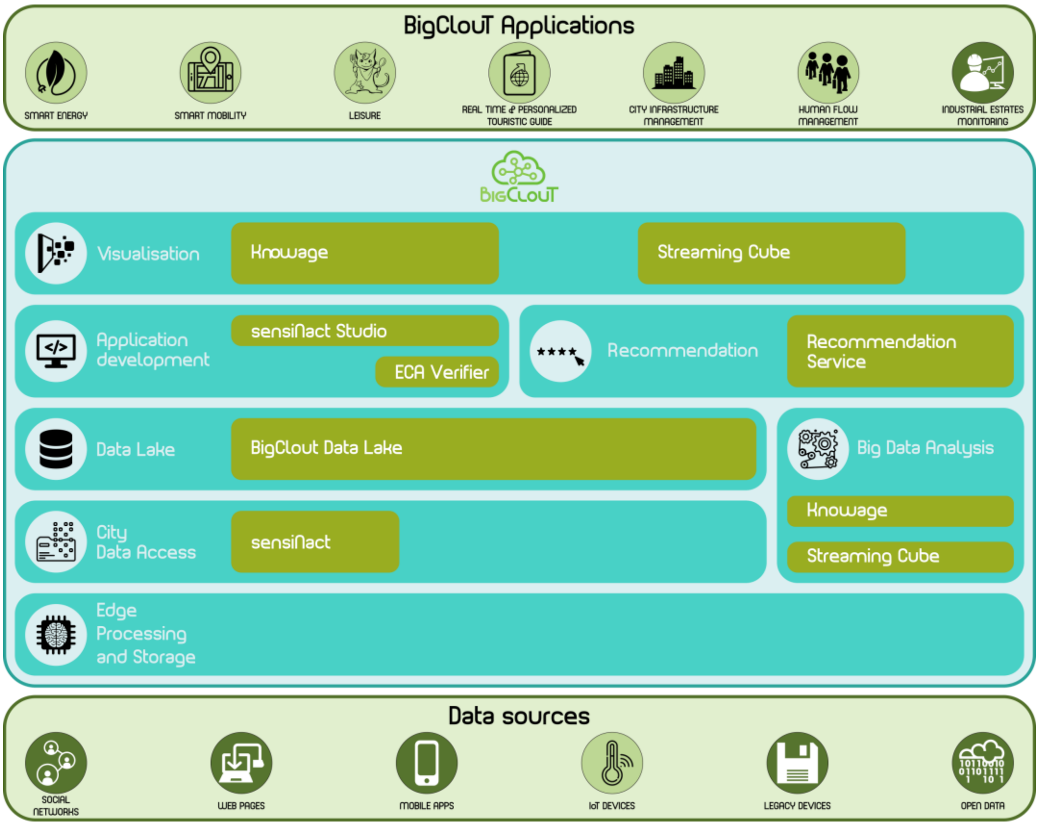
	Beginning	End		cSRL		
					Begining	end
BigClouT data lake	0,21	0,60		V	0,18	0,57
DeepOnEdge	0,07	0,52		L	1	2
D-NR platform	0,11	0,52				
D-NR studio	0,30	0,56				
Knowage	0,23	0,69				
Recommendation service	0,16	0,52				
SOXFire	0,16	0,57				

FIGURE 34 CITY INFRASTRUCTURE MANAGEMENT TRIAL IN FUJISAWA - OVERALL PICTURE AND SYSTEM READINESS LEVEL = 2 (TECHNOLOGY DEVELOPMENT)



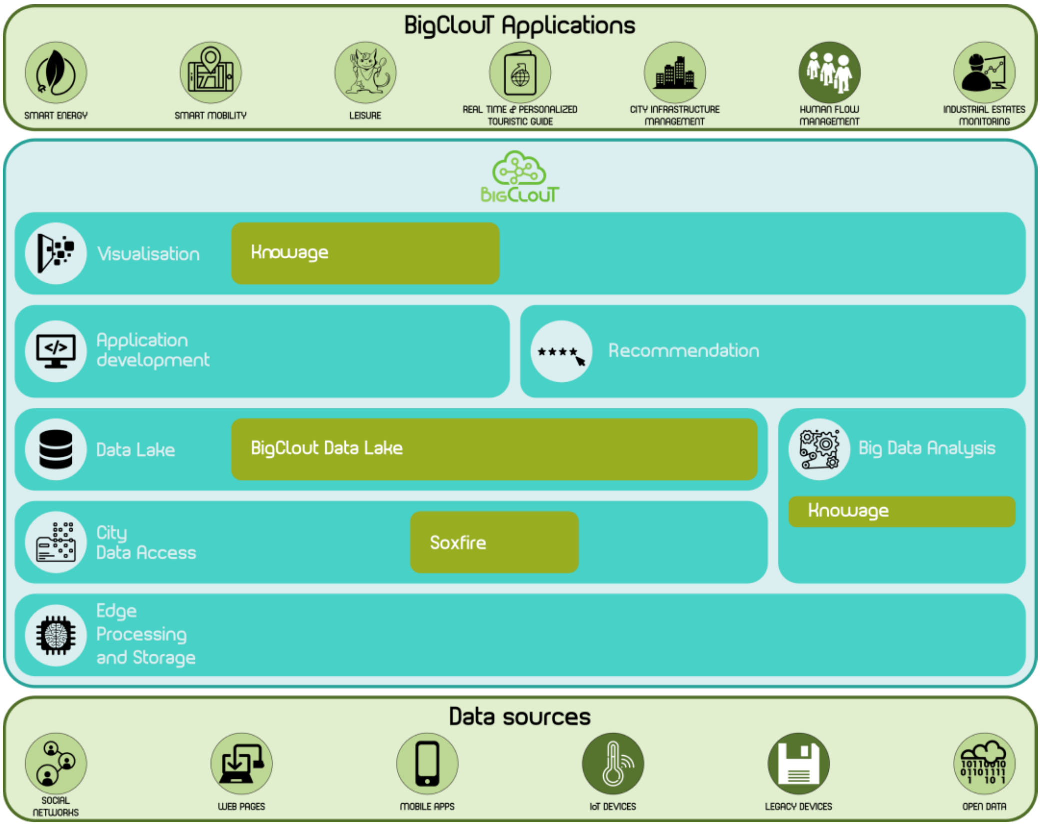
SRL				cSRL	
	Beginning	End		Begining	end
BigClouT data lake	0,25	0,63	V	0,20	0,61
Knowage	0,23	0,69	L	1	3
Recommendation service	0,16	0,52			
StreamingCube	0,16	0,61			

FIGURE 35 PERSONALISED TOURISTIC GUIDE TRIAL (HUKUREPO IN TSUKUBA) - OVERALL PICTURE AND SYSTEM READINESS LEVEL = 3 (SYSTEM DEVELOPMENT AND DEMONSTRATION)



SRL				cSRL	
	Beginning	End		Begining	end
BigClouT data lake	0,21	0,61	V	0,23	0,61
Knowage	0,23	0,69	L	1	3
Recommendation service	0,16	0,52			
SensiNact	0,24	0,60			
SensiNact Studio	0,41	0,64			
StreamingCube	0,13	0,58			

FIGURE 36 INDUSTRIAL ESTATES MONITORINGTRIAL IN GRENOBLE - OVERALL PICTURE AND SYSTEM READINESS LEVEL = 3 (SYSTEM DEVELOPMENT AND DEMONSTRATION)



SRL						
				cSRL		
	Beginning	End		Begining	end	
BigClouT data lake	0,33	0,74		V	0,29	0,76
Knowage	0,33	0,89		L	1	3
SOXFire	0,22	0,67				

FIGURE 37 HUMAN FLOW AND CITY MONITORING TRIAL (MINAREPO) IN FUJISAWA - OVERALL PICTURE AND SYSTEM READINESS LEVEL = 3 (SYSTEM DEVELOPMENT AND DEMONSTRATION)

4. BIBLIOGRAPHY

[1] BigClouT, “D1.1 Data collection tools and architecture”

[2] BigClouT, “D1.2 Citizen centric use cases and requirements”

[3] BigClouT, “D1.3 First BigClouT Architecture”

[4] BigClouT, “D1.4 Updated use cases, requirements and architecture”

[5] BigClouT, “D2.1 Data collection tools and architecture”

[6] BigClouT, “D2.2 Data collection and redistribution framework – demonstration”

[7] BigClouT, “D2.3 Self-aware distributed city data platform – first release”

[8] BigClouT, “D2.4 Distribtued data flow programming tool”

[9] BigClouT, “D2.5 Self-aware programmable city data platform – demonstration”

[10] BigClouT, “D2.6 Programmable Smart City – observations and lessons”

[11] BigClouT, “D3.1 Big data analytics framework architecture”

[12] BigClouT, “D3.2 Big data analytics framework report – first release”

[13] BigClouT, “D3.3 Big data analytics framework prototype”

[14] BigClouT, “D3.4 Big data analytics framework report – final release”

[15] BigClouT, “D4.3 Integrated use cases and final large scale deployments and experimentation

[16] BigClouT, “D4.5 Integrated use cases and final large-scale deployment and experimentation”

[17] Sauser, Brian & Verma, Dinesh & Ramirez-Marquez, Jose & Gove, Ryan. (0002). From TRL to SRL: The concept of systems readiness levels. Conference on Systems Engineering Research.

5. APPENDIX - TRL, IRL & SRL CALCULATION

In 1999, the Department of Defense of USA began implementing the Technology Readiness Level (TRL) as a metric to assess the maturity of a program's technologies before its system development begins. Despite the utility and value of the TRL as a metric for determining technology maturity before transitioning into a system, TRLs were not intended to address systems integration nor to indicate that the technology will result in successful development of a system. Integration Readiness Level (IRL) and System Readiness Level (SRL) provide new scales to address this gap. While TRL give an indication on components maturity, IRL gives an indication on the maturity of interactions between components and SRL provides an indication of maturity of the whole system (Figure 38).

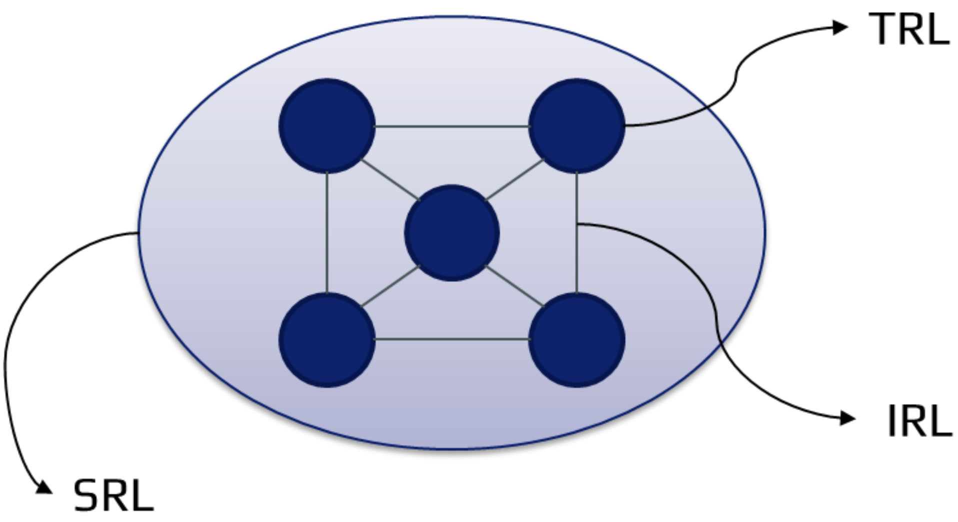


FIGURE 38: TRL, IRL & SRL

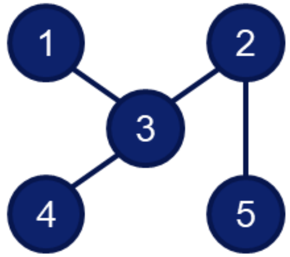
SRL is calculated from TRL and IRL. The steps are the following:

1. TRL and IRL matrix normalisation (all values are divided by 9)
2. Matrix product of the IRL matrix by the TRL vector
3. Division of each value of each vector value by the link number of the component plus one (it is considered that the component has a link with itself).
4. Sum of the values of the resulting vector
5. Division of the resulting value by the number of components in the system.
6. Assessment of the SRL level by comparing the value obtained at the limit of each level.

Figure 39 provides an example based on a five components system.

IRL	1	2	3	4	5
1	9	0	2	0	0
2	0	9	2	0	5
3	2	2	9	3	0
4	0	0	3	9	0
5	0	5	0	0	9

Block	TRL
1	3
2	4
3	6
4	5
5	7



Block	SRL
1	0,48
2	1,02
3	1,02
4	0,78
5	1,02

links + 1
2
3
4
2
2

SRLc = 0,35

Level	Name	SRL
1	Concept Refinement	0.1 → 0.39
2	Technology Development	0.4 → 0.59
3	System Development & Demonstration	0.6 → 0.79
4	Production & Development	0.8 → 0.89
5	Operations & Support	0.9 → 1.00

FIGURE 39: EXAMPLE OF SRL CALCULATION