



WP3 Extracting City Knowledge for Intelligent Services

D3.3 Big Data Analytics Framework Prototype – Demonstration

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: **ICCS** JP Editor: **TSU** Nature: Report

Dissemination: PU

Contractual delivery date: 2018-07-01

Submission Date: 2018-07-12



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

ABSTRACT

This deliverable consists in a technical report of the first version of the Big Data Analytics Framework. It describes the prototypes and the demonstrators of the outcomes of WP3 (Extracting City Knowledge for Intelligent Services) reported in Deliverable 3.2 (Big Data Analytics Framework report – first release).

Disclaimer

This document has been produced in the context of the BigClouT Project which is jointly funded by the European Commission (grant agreement n° 723139) and NICT from Japan (management number 18301). All information provided in this document is provided "as is" and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability. This document contains material, which is the copyright of certain BigClouT partners, and may not be reproduced or copied without permission. All BigClouT consortium partners have agreed to the full publication of this document. The commercial use of any information contained in this document may require a license from the owner of that information. For the avoidance of all doubts, the European Commission and NICT have no liability in respect of this document, which is merely representing the view of the project consortium. This document is subject to change without notice.



Revision history

Revision	Date	Description	Author (Organisation)
V0.1	08.05.2018	Initial ToC	Savong Bou, and Toshiyuki Amagasa (TSU)
V0.2	15.05.2018	Updated ToC	Orfefs Voutyras (ICCS)
V0.3	24.05.2018	Finalised ToC	Orfefs Voutyras (ICCS)
V0.4	01.06.2018	Section 6	Tadashi Okoshi (KEIO)
V0.5	06.06.2018	Section 4	George Palaiokrassas (NTUA)
V0.6	07.06.2018	Sections 1, 2, 6 & 8	Savong Bou, Toshiyuki Amagasa, and Hiroyuki Kitagawa (TSU)
V0.7	08.06.2018	Sections 3 & 5	Guiseppe Ciulla (ENG)
V0.8	21.06.2018	Section 2.1.3	Savong Bou, Toshiyuki Amagasa, and Hiroyuki Kitagawa (TSU)
V0.9	25.06.2018	Corrected references of figures	Toshiyuki Amagasa (TSU)
V0.10	28.06.2018	Section 4 final	Orfefs Voutyras (NTUA)
V0.11	03.07.2018	Merge, adjust mismatching figure reference, adjust reference	Savong Bou, Toshiyuki Amagasa, and Hiroyuki Kitagawa (TSU)
V0.12	05.07.2018	Modify based on the comments from CEA/NTTRD	Savong Bou, Toshiyuki Amagasa, and Hiroyuki Kitagawa (TSU)
V1.0	12.07.2018	Version ready for submission	Savong Bou, Toshiyuki Amagasa, and Hiroyuki Kitagawa (TSU)



TABLE OF CONTENTS

1	Introduction.....	8
1.1	Overview of D3.3.....	8
1.2	Objectives and goal of demonstration.....	9
2	Data Event Processing Prototype and Demonstration.....	10
2.1	JsSpinner/StreamOLAP.....	10
2.1.1	Demonstration.....	10
2.1.1.1	Description.....	10
2.1.1.2	Overview.....	10
2.1.1.3	Demonstrator.....	11
2.1.2	Package information and Installation instructions.....	13
2.1.3	User Manual.....	13
2.1.3.1	Supported Queries.....	13
2.1.3.2	Configuration Files.....	14
2.1.3.3	Compiling and Executing JsSpinner.....	14
2.1.4	Download-Source Code Repository.....	15
2.1.5	Licensing Information.....	16
3	Big Data Analysis.....	18
3.1	KNOWAGE Delivery and Usage.....	18
3.1.1	Demonstration.....	18
3.1.1.1	Overview.....	18
3.1.1.2	Analysis.....	18
3.1.2	Package information and Installation instructions.....	20
3.1.3	User Manual.....	20
3.1.4	Download – Source Code Repository.....	29
3.1.5	Licensing information.....	29
4	Machine Learning, Predictive Modelling and Decision making.....	31
4.1	Recommendation Service: Delivery and Usage.....	31
4.1.1	Demonstrator.....	31
4.1.2	Package information and Installation instructions.....	33
4.1.3	User Manual.....	35
4.1.4	Download - Source Code.....	36
4.1.5	Licensing information.....	37
5	Visualization.....	37
5.1	KNOWAGE.....	37
5.1.1	Demonstration.....	37
5.1.1.1	Demonstration – Cockpit.....	37
5.1.2	Package information and Installation instructions.....	39
5.1.3	User Manual.....	40
5.1.4	Download – Source Code Repository.....	45
5.1.5	Licensing information.....	46
6	Context Management and self-awareness.....	47
6.1	“Deep on Edge” Delivery and Usage.....	47
6.1.1	Demonstration – Demonstrator.....	47
6.1.2	Package information and Installation instructions.....	48



6.1.3	User Manual	48
6.1.4	Download – Source Code Repository	49
6.1.5	Licensing information	49
7	Conclusions.....	50
8	Bibliography	51



LIST OF FIGURES

Figure 1: City Data Processing Blocks involved in the demonstration	8
Figure 2: Overview of the demonstration	11
Figure 3: input query	12
Figure 4: output showing the amount of PM2.5 with respect to different areas	12
Figure 5: Source code package structure	15
Figure 6: KNOWAGE Demonstrator scenario	18
Figure 7: Fujisawa Road Damage Probability Dataset	19
Figure 8: Fujisawa Road Damage Probability Preview	19
Figure 9: KNOWAGE Home	21
Figure 10: KNOWAGE Menu	21
Figure 11: KNOWAGE Data Provider Menu Section	22
Figure 12: KNOWAGE Data Source	22
Figure 13: Dataset Creation Layout	23
Figure 14: Dataset Detail subsection	24
Figure 15: Dataset Type Selection	24
Figure 16: Dataset Advanced Functionalities	25
Figure 17: Example of Query Dataset	26
Figure 18: REST Dataset Definition	27
Figure 19: REST Dataset, Response Payload Management	28
Figure 20: CKAN Dataset definition	28
Figure 21: knowlage installation wizard	29
Figure 22: Example of a query to Neo4j web interface, retrieving nodes containing open data and user-generated data	31
Figure 23: Example of query to Neo4j web interface, retrieving nodes related to ratings and schedules, at a certain moment for all users	32
Figure 24: Example of query to Neo4j web interface, retrieving nodes related user preferences, based on the information they provided, while registering to the system	32
Figure 25: Example of a recommendation query for a heating schedules of a flat for the next hours	32
Figure 26: Prototype of our developed Graphical User Interface hosted in Okeanos Cloud Server. It provides quick access to end-users of the application	33
Figure 27: haproxy distributes requests across multiple neo4j servers aiming to optimize resource use, maximize throughput, minimize response time and avoid overload of any single resource	35
Figure 28: an example of templat	36
Figure 29: Fujisawa Demonstrator Cockpit	37
Figure 30: Fujisawa Cockpit - feature detail	38
Figure 31: Fujisawa Cockpit - Pie chart	38
Figure 32: Fujisawa Cockpit – Table	39
Figure 33: Fujisawa Cockpit - Export	39
Figure 34: KNOWAGE CE Visualization Framework	40
Figure 35: KNOWAGE Workspace Link	41
Figure 36: KNOWAGE Cockpit Toolbar	41
Figure 37: KNOWAGE Widget Panel	42
Figure 38: KNOWAGE Chart Widget Window	42
Figure 39: KNOWAGE Chart Structure	43
Figure 40: KNOWAGE Chart Configuration	43
Figure 41: KNOWAGE Advanced Chart Properties	44
Figure 42: KNOWAGE Map Widget Window	44



Figure 43: KNOWAGE Map Metadata Configuration.....	45
Figure 44: KNOWAGE Map Style Configuration.....	45
Figure 45: The captured road image with detected damage on the road markings.	47
Figure 46: The captured road image without any detected damage on the road markings.....	48



1 Introduction

1.1 Overview of D3.3

This deliverable is to describe the prototypes and the demonstrators of the outcomes of WP3 (Extracting City Knowledge for Intelligent Services) reported in Deliverable 3.2 (Big Data Analytics Framework report – first release). In the proposed architecture, *city data processing module* extracts city knowledge for intelligent services by analysing city data collected by *city resource access module* (Figure 1). The city data processing module is composed of different functionalities, namely, 1) data/event processing, 2) machine learning, and 3) big data analysis, being developed at different partners in BigClouT project, in such a way that different modules can interact with each other for achieving various types of analytics to fulfil the requirements of the project.

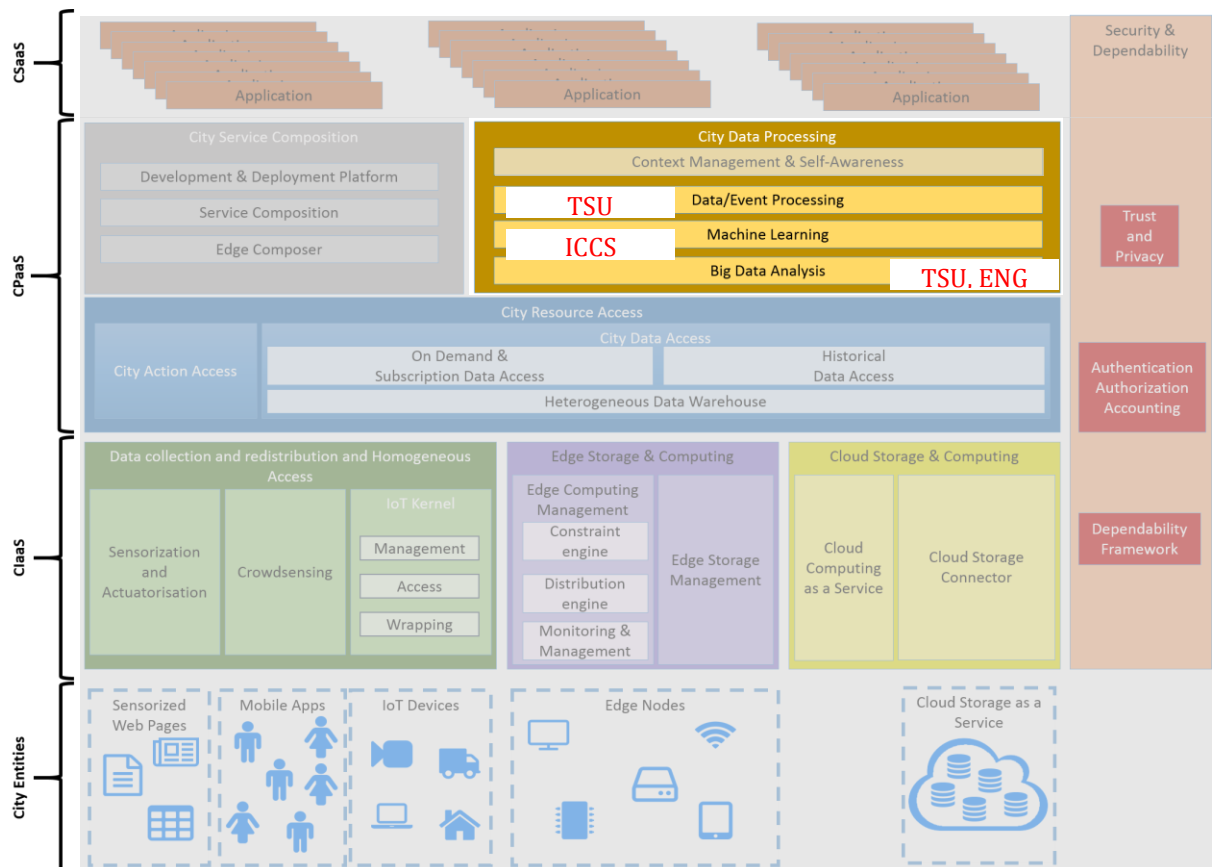


FIGURE 1: CITY DATA PROCESSING BLOCKS INVOLVED IN THE DEMONSTRATION

1.2 Objectives and goal of demonstration

This deliverable (D3.3) aims at presenting prototypes and demonstrators of the functionalities of the city data processing module. For this purpose, we take a modular approach where the prototype and demonstrator of each functionality is presented individually. In addition, we present an integrated demonstrator where different functionalities cooperate for achieving complicated analysis over city data. Unlike D3.2, this deliverable focuses on deployment and usage of each functionality whereas technical details, such as functional description and implementation details are provided in D3.2. To make it possible for readers to run the prototypes and demonstrators, we also provide necessary instructions of how to reuse the prototypes, i.e., download links, license information, installation instructions, package information, and user manuals. All demonstrations in this deliverable are summarized as follows:

1. **Section 2** demonstrates Data Event Processing functionality that collects environmental data in Fujisawa city via sensiNact gateway, and monitors the amount of PM2.5 in all villages in Fujisawa city by using both JsSpinner/StreamOLAP.
2. **Section 3** demonstrates Big Data Analysis functionality that deploys KNOWAGE to analyze the data about Fujisawa's road damage probability gathered daily by garbage trucks moving around the city.
3. **Section 4** demonstrates Machine Learning, Predictive Modeling and Decision making functionalities that introduce the recommendation service includes the delivery of a heating schedule to the households.
4. **Section 5** demonstrates Visualization that deploys KNOWAGE to visualize the information about Fujisawa's city road damages.
5. **Section 6** demonstrates Context Management and self-awareness functionality by deploying "Deep on Edge" to capture the video image of road condition with the degree of detection confidence.



2 Data Event Processing Prototype and Demonstration

2.1 [IsSpinner/StreamOLAP](#)

2.1.1 *Demonstration*

2.1.1.1 *Description*

Ambient air pollution has been closely associated with adverse health effects, such as cardiac and respiratory diseases. A recent research [1] has shown that high concentration of PM2.5 is associated with high blood pressure and cardiovascular events. Moreover, the study [2] has shown that there is an increased likelihood of morning hypertension, under the condition of high PM2.5 and low temperature.

This demonstration analyzes the environmental data in Fujisawa city and “Monitor the Amount of PM2.5 in all Villages in Fujisawa City” in real time. Fujisawa city is one of Japan’s most popular tourist spots. Her historical places and beautiful beaches attract a lot of visitors from Japan and around the world. Therefore, to guarantee the well-beings of all city citizens and visitors, closely monitoring the amount of PM2.5 in the air is very crucial. For example, at areas with high concentration of PM2.5, city staffs may be able to inform city citizens about the health risks, provide appropriate health preventions/aids, and ultimately find the root of causes and make the air quality to normality before serious health problem happens.

2.1.1.2 *Overview*

The overall picture of this demonstration is illustrated in Figure 2. At different areas of Fujisawa city, sensors are attached to garbage collecting cars. Those sensors send environmental information (location, PM2.5, temperature, pressure, altitude, wind speed, etc.) in real time to StreamOLAP system. StreamOLAP system applies various OLAP operations to analyse the received information, and sends the in-depth-analysed results to the administrative staffs in real time.



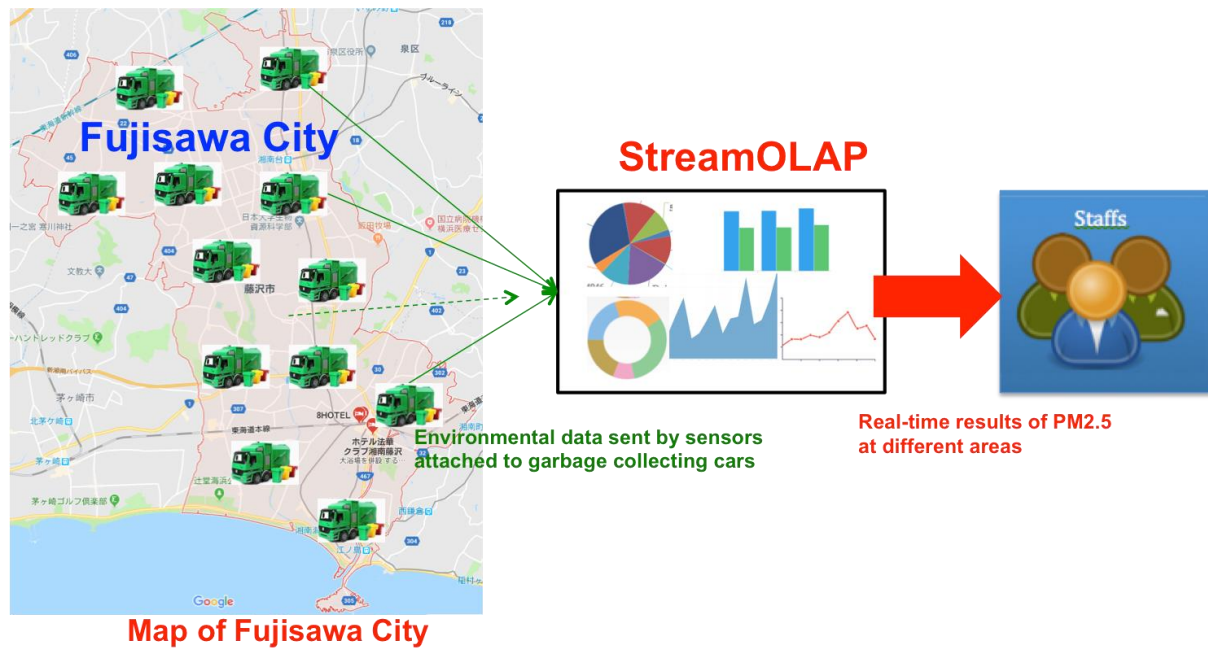


FIGURE 2: OVERVIEW OF THE DEMONSTRATION

2.1.1.3 Demonstrator

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 made accessible to StreamOLAP by using sensiNact gateway. In this demonstration, the following environmental data are used:

- Area name (village name)
- Temperature range (cool, chilly, warm , hot)
- PM2.5

Therefore, there are two-dimensional tables (Area and Temperature), which are useful for OLAP analysis.

Notice that, in this demonstration, the system accepts jaql-like query language [3]. A Web user interface of this system is provided so that users can submit queries to StreamOLAP system as shown in Figure 3.

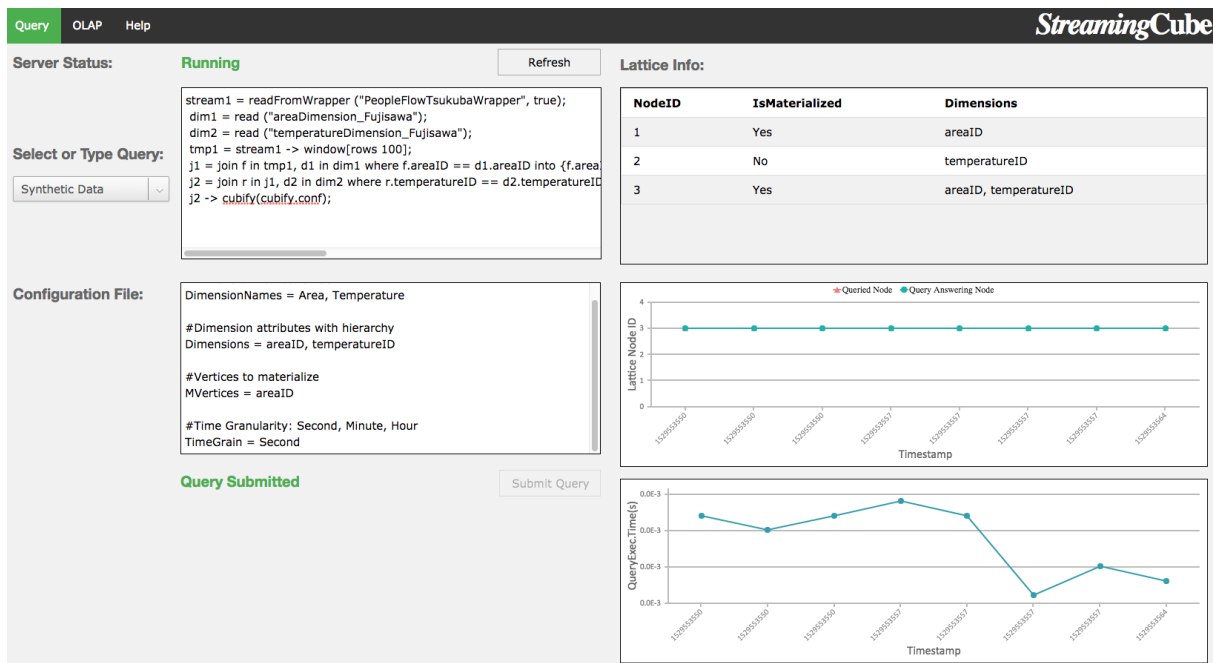


FIGURE 3: INPUT QUERY

After queries are submitted to the system, the Web user interface is used to extract results at various aggregation levels (e.g., roll up, drill down, sum, min, max). The results are in the forms of tables, charts, and graphs.

Figure 4 shows sample output results of the demo. As can be seen, the system provides useful options. For example, administrative staffs can choose to get the analysed results at dimension “areaID” or the combined dimensions of “areaID” and “temperatureID”.

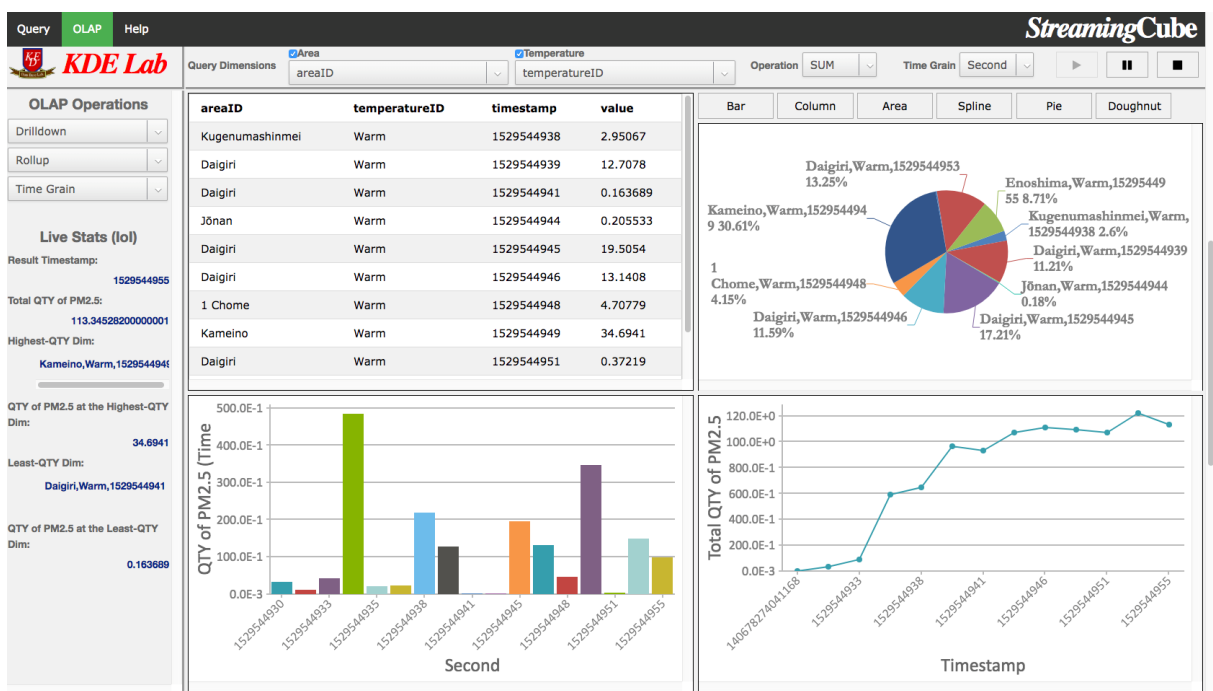


FIGURE 4: OUTPUT SHOWING THE AMOUNT OF PM2.5 WITH RESPECT TO DIFFERENT AREAS



2.1.2 Package information and Installation instructions

JsSpinner is a stream processing engine to process (filter, integrate, and/or aggregate) streams for subsequent data analysis. To run it, a Linux-box running Ubuntu is required. JsSpinner uses different packages. The instruction is as follows:

- Install required packages and libraries for Boost and g++:
 - apt-get install python-dev gccxml gzip
 - apt-get install xsltproc docbook-xsl docbook-xml
 - apt-get install build-essential g++ autotools-dev libicu-dev build-essential libbz2-dev
 - apt-get install zlib1g-dev
 - apt-get install libevent-dev
 - apt-get install curl libcurl3 libcurl3-dev php5-mcrypt php5-curl
- Get and install Boost libraries 1.49.0 with the following modifications:
 - Download boost libraries 1.49.0 "wget -c http://sourceforge.net/projects/boost/files/boost/1.49.0/boost_1_49_0.tar.gz/download"
 - Extract "tar zxvf boost_1_49_0.tar.gz"
 - Execute the following commands from the "boost_1_49_0" folder to remove the bug from boost libraries:
 - ~/Downloads/boost_1_49_0\$ grep -r "TIME_UTC" *
 - ~/Downloads/boost_1_49_0\$ find . -name ".*" -exec sed -i "s/TIME_UTC/TIME_UTC_/g" {} \;
 - Build the libraries ~/Downloads/boost_1_49_0\$./bootstrap.sh --with-libraries=all
 - Install the libraries ~/Downloads/boost_1_49_0\$ sudo ./b2 install
- JsSpinner requires our locally developed Query Parser to parse queries written in Jaql
 - Download the Query Parser and start it as follows before executing the JsSpinner
 - Install JDK if it is not available
 - sudo apt-get install openjdk-6-jdk
 - Install Eclipse
 - Import jsspinnercompiler-master project into Eclipse
 - Add the existing .Jar files into the project by right clicking on the jsspinnercompiler-master project->Properties->Libraries->Add JARS and adding antlr and gson .Jar files from the lib folder
 - Run the following two files from the specified folders to start the Query Parser
 - src/others/Run.java
 - src/test/JSSchemaServer.java

2.1.3 User Manual

2.1.3.1 Supported Queries

JsSpinner supports Continuous Queries (CQs) and Smart Continuous Queries (SCQ) while ordinary CQs are supported by most of the existing stream processing engines. Users can register Ordinary CQs to perform typical stream processing operations including select, filter, aggregate, join, group-by, etc.

- Sample CQ:



```

std::string query1 =

"stream1 = readFromWrapper (\\"performanceTestStream1\\", true) ;\

stream2 = readFromWrapper (\\"performanceTestStream2\\", false) ;\

tmp1 = stream1 -> window[range "+ stream1WindowSize +" seconds] ;\

tmp2 = stream2 -> window[rows "+ stream2WindowSize +"] ;\

j = join s in tmp1, d in tmp2 where s.A == d.A into {s.B, d.C} ;\

j -> istream;" ;

```

2.1.3.2 Configuration Files

JsSpinner requires one configuration file named "JStreamSpinner.conf". Sample configuration files along with the description of required parameters are available in the config folder of the JsSpinner code.

2.1.3.3 Compiling and Executing JsSpinner

- Compilation
 - jssspinner/jsonstream\$ make
- Execution
 - jssspinner/jsonstream\$./main.exe

Once under execution, JsSpinner/StreamingCube can accept queries (CQs/SCQs) through the ports specified in the configuration file.



2.1.4 Download-Source Code Repository

The source code can be downloaded from the following link.

<http://www.streamspinner.org/streamingcube/download.html>

Inside the download folder, JsSpinner is in directory "jsonstream". Inside that directory, there are four main components as shown in Figure 5:

1. src:
Is a folder that contains all source codes. The details are explained in next section
2. main.exe:
Is the executable file.
3. Makefile:
The system is compiled by using the Makefile. When compiling, if more dependent libraries are required, just add the full paths of those libraries into "LIBDIR" and "INCLUDES".
4. configure:
Is a folder that contains all configuration files of the system.

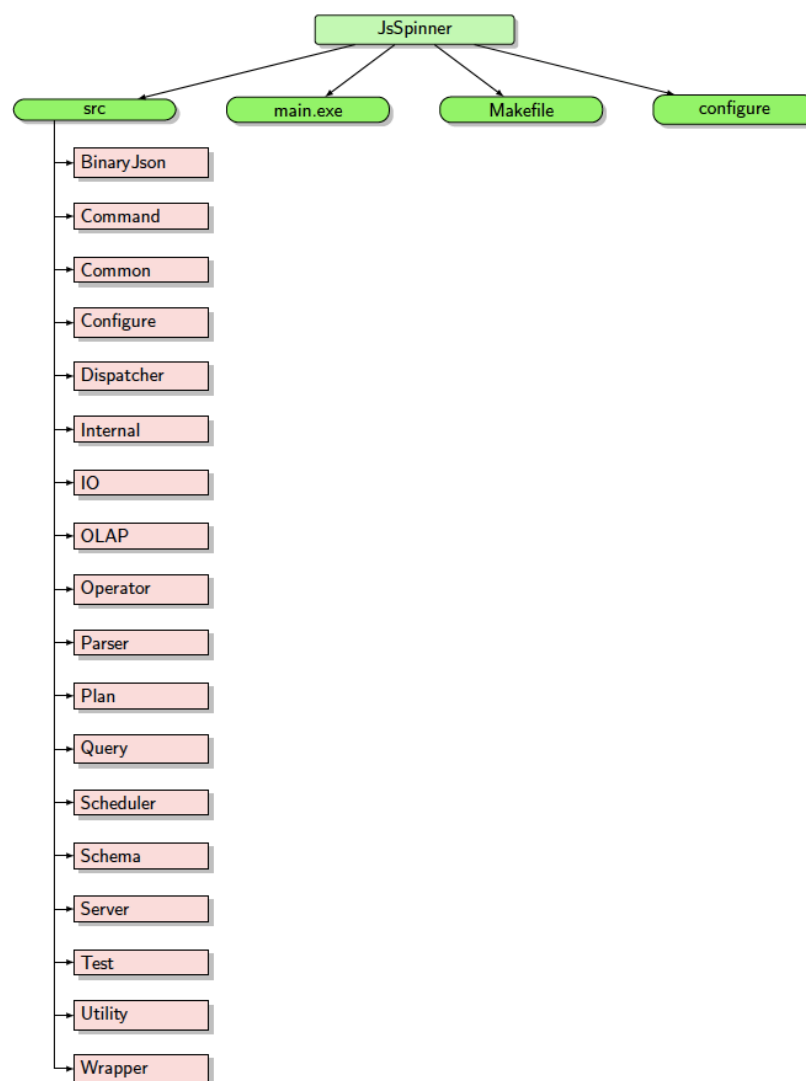


FIGURE 5: SOURCE CODE PACKAGE STRUCTURE

The source folder **src** contains the sub-folders as shown in Figure 5. Each sub-folder contain various classes. The main tasks of each sub-folder are summarized as follow:

- **BinaryJson:**
Maintain JSON specifications that are supported by the system.
- **Command:**
Is responsible for accepting the commands, and triggering the system based on the accepting commands.
- **Common:**
Maintain the management of the types of JSON.
- **Configure:**
Maintain all configuration files, which are required for the initialization of the system.
- **Dispatcher:**
Is responsible for accepting input from wrapper and send to the system.
- **Internal:**
Allocate and manage memory for queues and synopsis.
- **IO:**
Create synthetic input data streams or accept the input data streams.
- **OLAP:**
Maintain main processing of StreamOLAP.
- **Operator:**
Maintain all supported operators (i.e., LeafOpeartor, RootOperator, JoinOperator,...) of the system.
- **Parser:**
Manage the input and output of the registered queries.
- **Plan:**
Generate query plan and instantiate operators by query intermediate representation.
- **Query:**
Parse the input queries and register them to the system. It is also responsible for managing the input and output of the registered queries.
- **Scheduler:**
Maintain information about the processing orders of all operators.
- **Schema:**
Register schema information.
- **Server:**
Is responsible for initializing all modules in the system, which initializes the JsSpinner/StreamOLAP. It is also responsible for processing all operators against the incoming streams.
- **Test:**
Contain the main class of the system. The input query is defined here.
- **Utility:**
Manage the utilizations of CPU, generation of timestamp, and other system's statuses.
- **Wrapper:**
Read the wrapper specification in the wrapper folder, and register each wrapper. It is also responsible for parsing the input data streams into elements recognized by the system.

2.1.5 Licensing Information



Currently JsSpinner is a copyright © 2018 product and is distributed by Kitagawa Data Engineering Laboratory, University of Tsukuba under the Apache License, Version 2.0. For more information, referring to the detail information following the link

<http://www.streamspinner.org/streamingcube/documentation.html>



3 Big Data Analysis

3.1 KNOWAGE Delivery and Usage

3.1.1 Demonstration

This section of the document describes a possible scenario that is currently under investigation about Big Data Analytics exploiting KNOWAGE's capabilities. It is important to underline that this scenario is not intended to be the final one and it might be susceptible to changes.

3.1.1.1 Overview

The aim of this demonstration is to analyse data about Fujisawa's road damage probability gathered daily by garbage trucks moving around the city. The output of the analysis will be an ordered list of the most damaged areas in the city that will be used by Fujisawa's municipality in order to schedule the proper controls or maintenance services. As a result, the analysis will produce information about the most damaged areas in the city. Figure 6 illustrates the global schema of the demonstration.

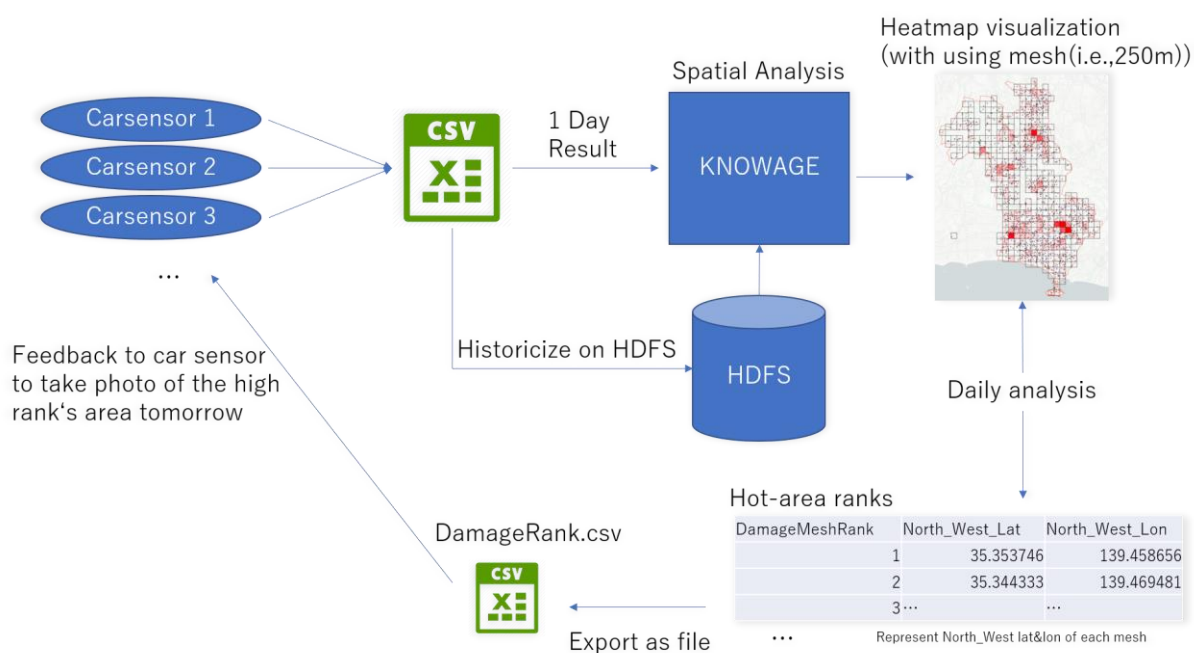


FIGURE 6: KNOWAGE DEMONSTRATOR SCENARIO

Data gathered daily by the garbage trucks will be provided as csv file and stored in an HDFS based infrastructure for future historical analysis. The aim is to connect this HDFS infrastructure with KNOWAGE in order to perform analysis on historical data. The specific HDFS platform to be used is currently under investigation.

3.1.1.2 Analysis

At the time this document is written, a small subset of the data is used. This dataset contains the damage probability measurements of a single day and is composed by the following fields:

- time: the time of the day the measurement was taken
- carnum: the identifier of the garbage truck
- latitude: the latitude of the measurement
- longitude: the longitude of the measurement
- damage_probability: the damage probability in the specific location taken by the garbage truck in the specific time of the day

The data is provided in form of csv file and then loaded into KNOWAGE's server through the File dataset type. Figure 7 depicts the dataset loaded into KNOWAGE's server and Figure 8 shows the preview of the data.

DETAIL TYPE ADVANCED

DataSet Type *
File

Uploaded file: FujisawaDamageProbability.csv CHANGE FILE

Delimiter Character * , Quote Character * " Encoding UTF-8 Date Format yyyy-MM-dd

NEXT >

FIGURE 7: FUJISAWA ROAD DAMAGE PROBABILITY DATASET

time	carnum	latitude	longitude	damage_probability
08:10	40	35.353746666666666	139.45865666666666	0.38344653624679614
08:10	55	35.35364333333333	139.45881666666668	0.6888241484492722
08:10	53	35.35359833333333	139.45887833333333	0.8150703868744607
08:10	50	35.37773	139.44647166666667	0.18733517379080056
08:10	22	35.377671666666664	139.44662333333332	0.4768337358488468
08:10	30	35.334615	139.49513666666667	0.7672625899990659
08:10	26	35.33658666666667	139.49696333333333	0.6332271210744357

FIGURE 8: FUJISAWA ROAD DAMAGE PROBABILITY PREVIEW

Two different analysis are currently under development over this dataset. In both cases, the aim is to divide Fujisawa's city area into sub-areas by using a square grid and get the most critical area in the city. Dividing the overall city area into a grid allows to group the measurements taking into

account the specific feature of the grid each measurement falls into. In particular, the analysis under development are:

1. The first analysis will show how many measures in a city area exceed a threshold (e.g. 0.5). It takes the provided measurements as input and will estimate the most critical areas in the city.
2. The second analysis will estimate the status of each city area. The output of this analysis will suggest which city area has the highest damage probability. The estimation will be evaluated by calculating the number of measurements grouped into the following categories:
 - Low Damage Risk: if most of the measurements does not exceed the value of 0.3.
 - Medium Damage Risk: if most of the measurements does not exceed the value of 0.6.
 - High Damage Risk: if most of the measurements exceed the value of 0.6.

Moreover, a specific numerical value will be evaluated and used to provide the list of the most critical areas in the city.

3.1.2 Package information and Installation instructions

KNOWAGE is available in two versions: the Community Edition (KNOWAGE CE) and the Enterprise Edition (KNOWAGE EE); KNOWAGE CE is freely available, whereas KNOWAGE EE is available only by subscription.

KNOWAGE CE installation packages can be downloaded from a dedicated web pages [4]; it requires at least 2 GB of free space on file system and 3 GB of RAM; installation packages are available for Windows and Linux-Ubuntu OS; to properly install and run KNOWAGE CE it is necessary to have previously installed Java SE Development Kit 8u161 [5] and MySQL Server 5.5 [6], or alternatively MariaDB Server 10.2 [7]. More details are available at [8].

This section, as well as section 5.1, refers to the KNOWAGE CE version of the suite.

3.1.3 User Manual

KNOWAGE's main point of access is its GUI. After a successful login, the user will access the home page depicted in Figure 9.





FIGURE 9: KNOWAGE HOME

He/she can access the functionalities by clicking on the top left button and the main menu will be visible, Figure 10.

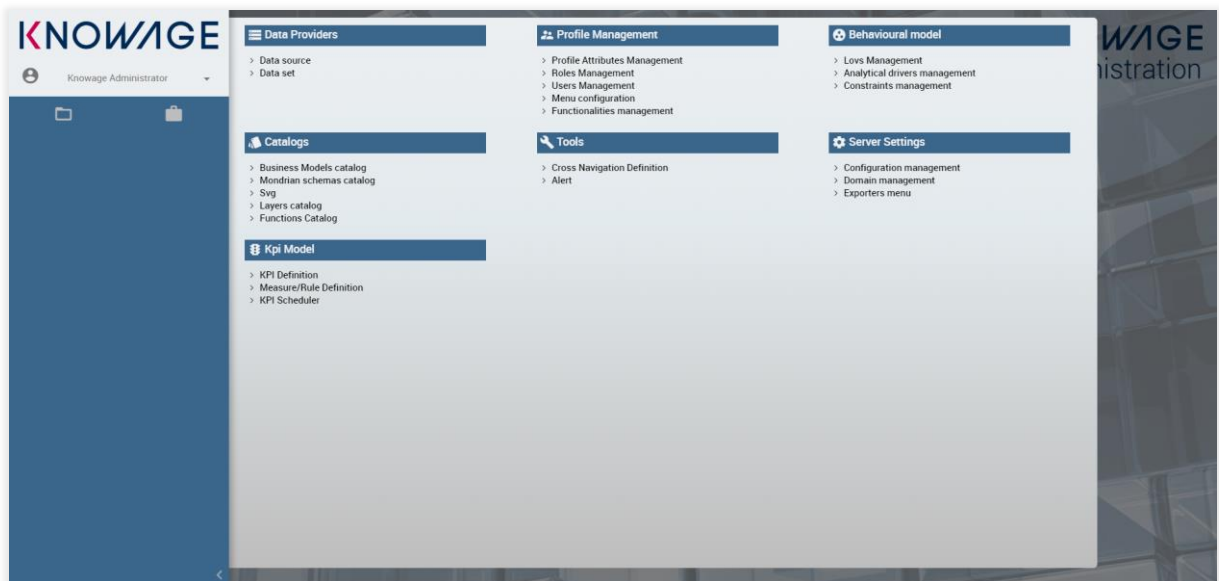


FIGURE 10: KNOWAGE MENU

This section of the document covers the steps needed to:

- Create a new Data Source
- Create a new Dataset

By creating a Data Source, the user will be able to use its data in analysis through the definition of a Datasets. Indeed, a Dataset is the portion of data used to perform analysis. Datasets can be retrieved by querying the previously defined Data Sources or from external data providers.

Create a new Data Source



In order to create a new Data Source, the user should first check if the proper java driver is correctly deployed into the application server. For instance, in case of a MySQL data source and Apache Tomcat [9] as application server, the user should check for the presence of the “*mysql-connector-java-<VERSION>-bin.jar*” into the folder “<path_to_tomcat>/lib”. This first step is mandatory, since the connection to those type of Data Sources cannot be established, if the driver is not deployed.

The user should now open the Data Sources management page by clicking on the proper link in the main menu under the *Data Providers* section, as illustrated in Figure 11.

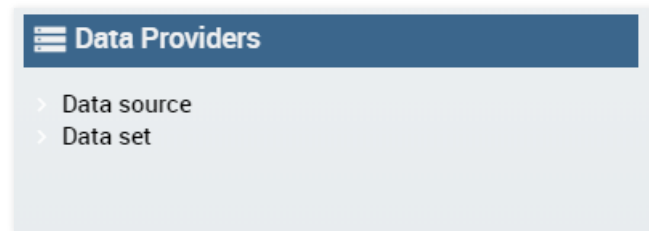


FIGURE 11: KNOWAGE DATA PROVIDER MENU SECTION

The Data Source management page lists the existing data sources in the right panel and by clicking on the plus button a new Data Source can be defined. Figure 12 shows the Data Source page layout.

Label	Description
ds_cache	KNOWAGE_CACHE
ds_cache_jdbc	test
Foodmart	Foodmart
Lufdaten_MongoDB	Lufdaten_MongoDB

Form fields for creating a new Data Source:

- Label * (Required)
- Description
- Dialect * (Required)
- ☐ Multischema
- Read Only: ☒ Read Only, ☐ Read and write
- Type: ☒ JDBC, ☐ JNDI
- URL * (Required)
- User
- Password
- Driver * (Required)

FIGURE 12: KNOWAGE DATA SOURCE

In order to properly define the Data source, the user must specify its dialect and a unique label. Moreover, the user should specify if the data source should be *Read Only* or *Read and Write*. In case of *Read Only* data sources, additional temporary tables will not be written in the data source. In case of *Read and Write* data sources KNOWAGE's temporary tables will be written in the data source. The user must then select the type of the Data Source choosing between JDBC and JNDI and enter the required information. Specifically, by selecting JDBC the user has to insert the URL and the driver and the username and password, if authentication is required. By selecting JNDI, the user should specify the connection string defined into the application server.

The user should then test the connection by clicking on the *TEST* button and save the new Data Source if test is successful.

Create a new Dataset

In order to create a new Dataset, the user should click on the proper link in the Data Provider section of the main menu, Figure 11. The Dataset management page (Figure 13) lists the existing datasets in the right panel and by clicking on the plus button a new Dataset can be defined.

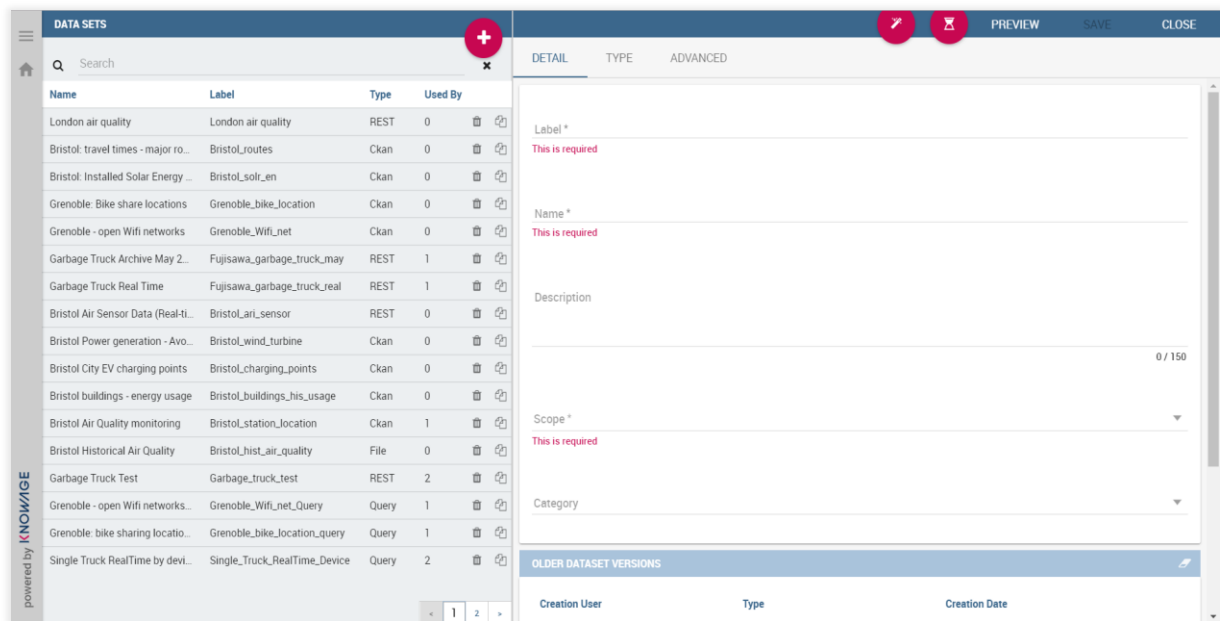


FIGURE 13: DATASET CREATION LAYOUT

The Dataset creation widget is divided in three subsections:

- Detail:** which allows the user to insert some metadata about the new Dataset (Figure 14: Dataset Detail subsection). It is mandatory to define a unique label and a name for the new Dataset. Moreover, it is mandatory to select a scope for the Dataset which can assume one of the following values:
 - User: which limits the scope of the dataset to the user who's creating it.
 - Enterprise: which grants the visibility of the dataset to every users.
 - Technical: which limits the visibility of the dataset to technical users.

DETAIL TYPE ADVANCED

Label *

This is required

Name *

This is required

Description

0 / 150

Scope *

This is required

Category

FIGURE 14: DATASET DETAIL SUBSECTION

- Type: which allow to select the type of the Dataset (Figure 15) and to configure its specific properties.

DETAIL TYPE ADVANCED

DataSet Type *

File

Query

Java Class

Script

Qbe

FIGURE 15: DATASET TYPE SELECTION

- **Advanced:** which provides advanced operation on dataset, Figure 16. For instance, this section provides, among other functionalities, the possibility to persist the dataset in KNOWAGE's internal cache.

The screenshot shows a web interface for configuring a dataset. At the top, there are four red circular icons: a key, a lock, a timer, and a question mark. To the right of these icons are buttons for 'PREVIEW', 'SAVE', and 'CLOSE'. Below this is a tabbed interface with three tabs: 'DETAIL', 'TYPE', and 'ADVANCED'. The 'ADVANCED' tab is selected. The main content area is divided into three sections. The first section contains 'Transformation Type: PIVOT_TRANSFORMER' with an unchecked checkbox. The second section contains 'Exportable on HDFS:' with an unchecked checkbox. The third section contains 'Persist:' with a checked checkbox. Below this, there is a red error message: 'Table name * This is required'.

FIGURE 16: DATASET ADVANCED FUNCTIONALITIES

KNOWAGE supports, at the time this document is written, several dataset types:

- **Query:** performing a query over an already defined Data Source, following its dialect.
- **Flat:** this dataset type is also related to Data Sources. This dataset type will contain a complete table of a selected Data Source.
- **Query by Example (QbE):** this dataset type is retrieved from a Data source whose schema was used to create a Business Model. The Business Model creation is out of the scope of this document.
- **File:** by selecting this type, the dataset will contain the data coming from a csv or an xls file uploaded by the user.
- **Java Class:** the dataset is retrieved from a java class loaded into KNOWAGE's application server
- **Script:** the dataset is retrieved from a Javascript or a Groovy script.
- **Ckan:** the dataset is retrieved from a specific resource in a CKAN repository.
- **REST:** the dataset is retrieved from a RESTful API.
- **Solr:** the dataset is retrieved by querying a Solr endpoint.
- **SPARQL:** the dataset is retrieved by querying a SPARQL endpoint

In this section will be provided examples for Query, CKAN and REST datasets.

In order to create a Query dataset a Data Source is mandatory since the query will be executed over an existing data source. As a matter of fact, by selecting the Query type from the list depicted in Figure 15, the user has to select a Data Source and he has to write a query following its specific dialect. Moreover, he can define parameters in the query by using KNOWAGE's specific syntax

$\$P\{parameter_name\}$ and configure it in the parameters section of the form. Figure 17 illustrates an example of query over a MySQL data source using parameters.

DETAIL TYPE ADVANCED

DataSet Type *
Query

Data Source *
ds_cache

Query

```

1 select id, device, pm2_5, pressure, speed, temperature, illuminance, longitude, latitude, location, altitude,
2 DATE_FORMAT(CONVERT_TZ(from_unixtime(timestamp), @@session.time_zone, 'Japan'), "%Y-%m-%d %T") as date, timestamp
3 from garbage_truck_realtime
4 where device=$P{device} and pm2_5 > 0 and timestamp <= UNIX_TIMESTAMP(date_add($P{data_in}, INTERVAL 24 HOUR)) and
5 timestamp >= UNIX_TIMESTAMP($P{data_in})
6 order by date desc

```

EDIT SCRIPT

PARAMETERS

Name	Type	Default Value	Multivalue
device	String	carsensor010	<input type="checkbox"/>
data_in	String	2018-01-15	<input type="checkbox"/>

FIGURE 17: EXAMPLE OF QUERY DATASET

REST Dataset does not need the definition of any Data Sources. By selecting REST from the list of types in the list at Figure 15 it is mandatory to insert the url of the services and its http method and a request body if needed. Optionally, the user should insert request headers in the proper section, Figure 18.

DETAIL
TYPE
ADVANCED

DataSet Type *
REST

Address *
This is required

Request body
0 / 2000

HTTP methods *
This is required

REQUEST HEADERS

Name	Value
------	-------

FIGURE 18: REST DATASET DEFINITION

Moreover, the user should choose if using the JSONPath notation to access the payload or in case the REST service is NGSI compliant he can select the NGSI option, Figure 19.

JSON Path Items

Use directly JSON Attributes: ☐

NGSI: ☐

Name	JSON path value	Type or JSON path type
------	-----------------	------------------------

FIGURE 19: REST DATASET, RESPONSE PAYLOAD MANAGEMENT

In order to define a CKAN Datasets, the URL of a CSV or XLS resource belonging to the CKAN repository should be specified. The user should insert specific CSV or XLS properties such as the delimiter or the encoding for the CSV type. It is mandatory to insert the CKAN id of the resource and its URL. Figure 20 depicts the form for the CKAN CSV type.

DETAIL TYPE ADVANCED

DataSet Type *

Ckan

File type *

CSV

Delimiter Character * This is required

Quote Character * This is required

Encoding This is required

Date Format This is required

CKAN Id * This is required

CKAN Uri * This is required

FIGURE 20: CKAN DATASET DEFINITION

In order to properly store a Dataset, the user must visualize its preview by clicking the proper button. The preview helps the user to validate the dataset. Moreover, the user should manage the Dataset field's metadata by clicking the proper button. Dataset's fields' metadata could assume one of the following values:

- **Attribute:** the field can be used as a label or a grouping property in analytical documents.
- **Measure:** the field would be used as value to be analysed into the documents.
- **Spatial Attribute:** the field would be used as input for map widgets; this widget is currently in an early stage and it will be released in next versions of KNOWAGE CE [10].

Additional and more detailed information about the usage of KNOWAGE can be found in [11].

3.1.4 Download – Source Code Repository

Installation packages of KNOWAGE CE (current release 6.2) are available at [4], whereas its source code is available on a dedicated GitHub page [12]. The installation of the tools takes advantage of an interactive wizard, Figure 21, where the user has to select the installation folder and the default database to be used. For instance, an inner MariaDB or an external MySQL server. By selecting MariaDB no further operations are needed. By selecting MySQL, the user has to specify the server URL, port, username and password. The installer will then deploy the framework using Tomcat as application server creating two databases into the selected server. These databases are, respectively, *knowage_ce* which is the main database used by KNOWAGE and a demo database named *Foodmart*.

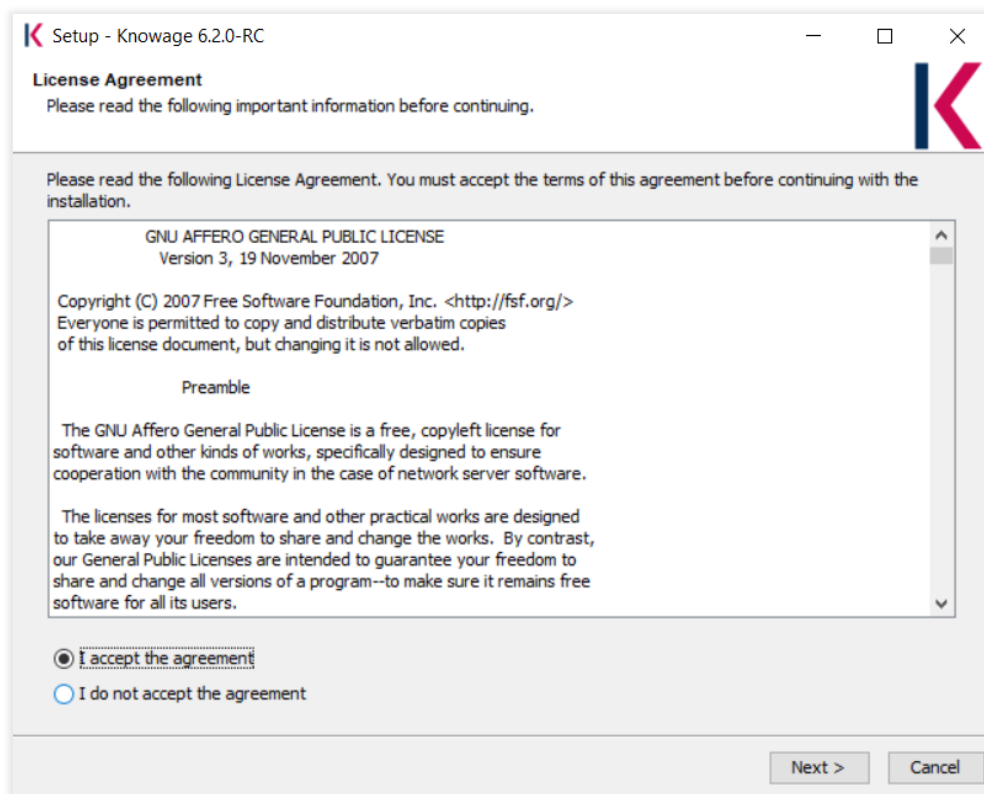


FIGURE 21: KNOWAGE INSTALLATION WIZARD

3.1.5 Licensing information



Source code of KNOWAGE CE is released under the terms of GNU Affero GPL v3 [[13](#)]



4 Machine Learning, Predictive Modelling and Decision making

4.1 Recommendation Service: Delivery and Usage

4.1.1 Demonstrator

Examples of end-user queries

An indicative use case of our recommendation service includes the delivery of a heating schedule, which will be applied to a flat. The user of the application receives the top scheduling recommendations and chooses one to be automatically applied for the following hours of interest. For example, a resident of a 60 m2 flat in the Camden borough with northern orientation wishes to receive an economical heating for the following ten hours. Using the web user interface provided by Neo4j, the user of our application could perform queries as shown below.

1. Visualization of open data regarding external weather conditions and temperature inside flats for a specific time.

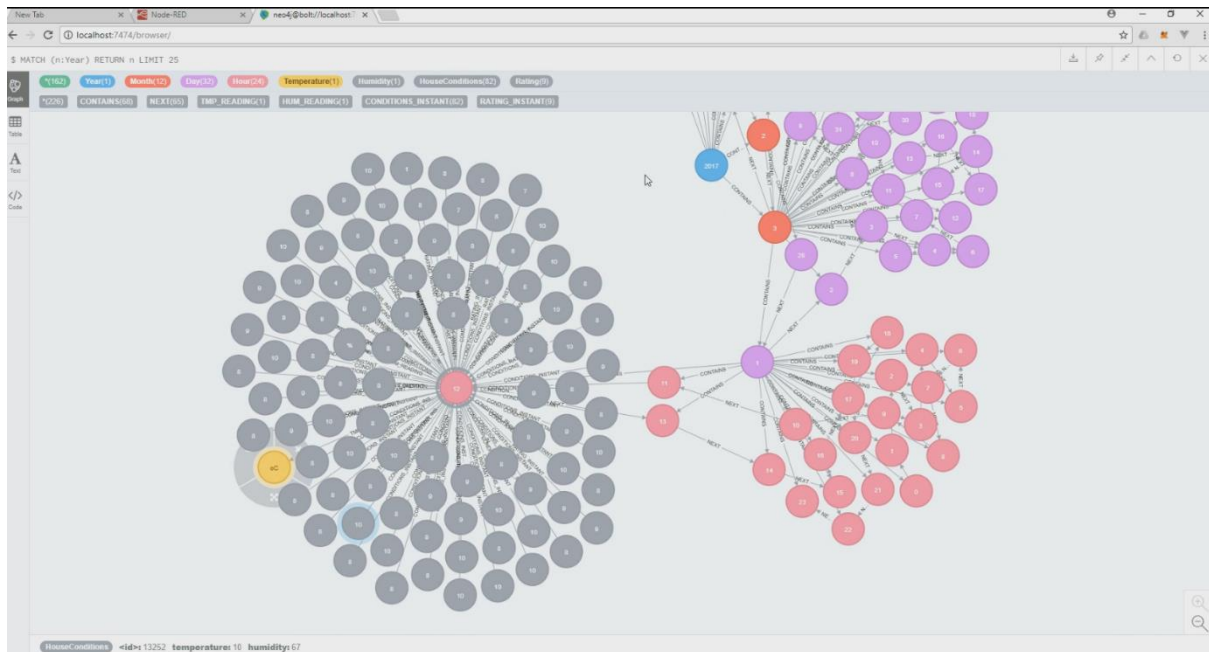
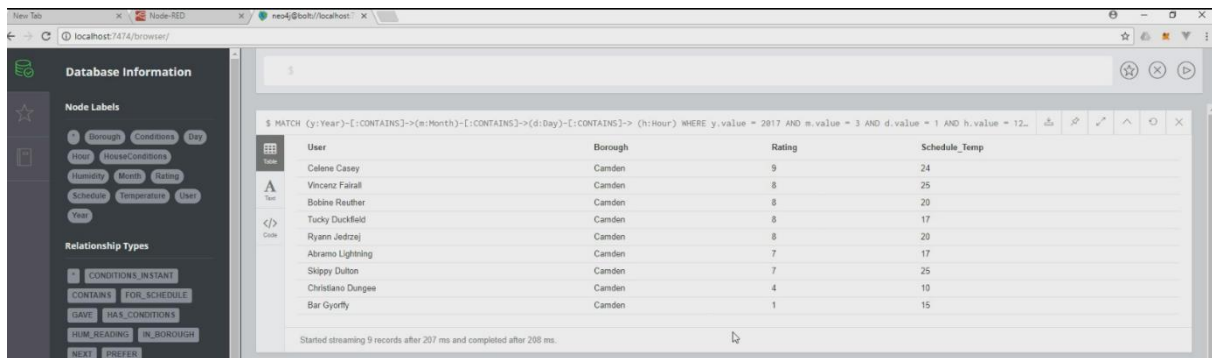


FIGURE 22: EXAMPLE OF A QUERY TO NEO4J WEB INTERFACE, RETRIEVING NODES CONTAINING OPEN DATA AND USER-GENERATED DATA

2. Query to retrieve ratings and schedules, at a certain moment for all users



```

MATCH (y:Year)-[:CONTAINS]->(m:Month)-[:CONTAINS]->(d:Day)-[:CONTAINS]->(h:Hour) WHERE y.value = 2017 AND m.value = 3 AND d.value = 1 AND h.value = 12

```

User	Borough	Rating	Schedule_Temp
Celine Casey	Camden	9	24
Vincenz Fairall	Camden	8	25
Bobine Reuther	Camden	8	20
Tucky Duckfield	Camden	8	17
Ryann Jedrej	Camden	8	20
Abramo Lightning	Camden	7	17
Skippy Dulton	Camden	7	25
Christiano Dungee	Camden	4	10
Bar Gyorffy	Camden	1	15

FIGURE 23: EXAMPLE OF QUERY TO NEO4J WEB INTERFACE, RETRIEVING NODES RELATED TO RATINGS AND SCHEDULES, AT A CERTAIN MOMENT FOR ALL USERS

3. Query to retrieve user preferences for a specific user



```

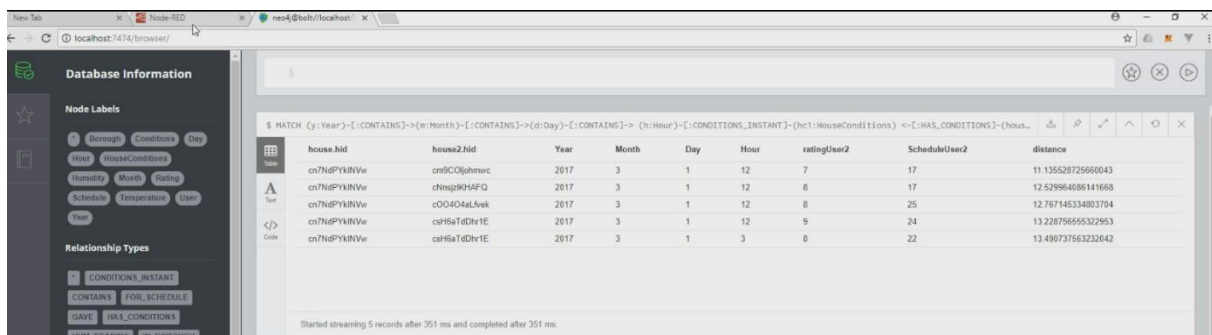
MATCH (u:User {hid: "cn7NpYkNVw"}) -[:PREFER]->(c:Conditions) RETURN u.hid AS hid, u.name AS Name, c.desiredtemp AS desiredtemp, c.desiredhumid...

```

hid	Name	desiredtemp	Humidity	Illness
cn7NpYkNVw	Christiano Dungee	17	70	FALSE

FIGURE 24: EXAMPLE OF QUERY TO NEO4J WEB INTERFACE, RETRIEVING NODES RELATED USER PREFERENCES, BASED ON THE INFORMATION THEY PROVIDED, WHILE REGISTERING TO THE SYSTEM

4. Recommendation based on orientation, surface, desired temp, desired humidity, consumer profile, current temperature, current humidity and returning schedule



```

MATCH (y:Year)-[:CONTAINS]->(m:Month)-[:CONTAINS]->(d:Day)-[:CONTAINS]->(h:Hour)-[:CONDITIONS_INSTANT]->(hc1:HouseConditions) <-[:HAS_CONDITIONS]->(hou...

```

house.hid	house2.hid	Year	Month	Day	Hour	ratingUser2	ScheduleUser2	distance
cn7NpYkNVw	cnSC0lphmnc	2017	3	1	12	7	17	11.135528725660043
cn7NpYkNVw	cnmp28HAFQ	2017	3	1	12	8	17	12.529964086141668
cn7NpYkNVw	coO4O4eLvck	2017	3	1	12	8	25	12.767145334803704
cn7NpYkNVw	caH5aTdDhr1E	2017	3	1	12	9	24	13.2287565532953
cn7NpYkNVw	caH5aTdDhr1E	2017	3	1	3	8	22	13.490737563232042

FIGURE 25: EXAMPLE OF A RECOMMENDATION QUERY FOR A HEATING SCHEDULES OF A FLAT FOR THE NEXT HOURS

Our Service exposes a RESTful web service, which allows the integration with other tools, services. Additionally, we have implemented a web interface, so end-users are able to connect to the service. This way, we could perform queries and retrieve a recommendation for the supported use cases.

This interface serves the needs of Neo4J users for the BigCloudT platform.
It has three main functions:
(a) "Preferences for a specific user",
(b) "Ratings for a specific date and time" and
(c) "Recommendations".
Option (a) will display the preferences of the user which has the Name provided in the form.
Option (b) will display the preferences of the users for the specified day and time.
Option (c) will display a ranked set of recommendations, based on both the Name, the day and the time specified in the form, by comparing the choices of the other users registered in our database.

Function		Query Result								
Name	HouseID	Year	Month	Day	Hour	Rating	Schedule	Distance		
Bobine Reuther	sPrxmdZHLc9O	20	3	1	12	8	20	10.34		
Neil MacConchie	9tB0MMpocCbK	20	3	1	15	1	30	10.63		
Christiano Dungee	wk7NYNcndVPI	20	3	1	12	4	10	13.22		
Ainslee Balassa	s3a9VkJZSiqck	20	3	1	14	8	30	13.74		
Abramo Lightning	omj19OCcmhw	20	3	1	3	6	14	14.83		

FIGURE 26: PROTOTYPE OF OUR DEVELOPED GRAPHICAL USER INTERFACE HOSTED IN OKEANOS CLOUD SERVER. IT PROVIDES QUICK ACCESS TO END-USERS OF THE APPLICATION.

4.1.2 Package information and Installation instructions

- **Operating System:** We have tested the platform on Windows 10 and Ubuntu 18 but all of the software listed here is available in a large number of other distributions.
- **Port Forwarding:** Both on Windows and Ubuntu we have to allow traffic to ports 80 for the web user interface, 1880 for the node-red and 7474 for the Neo4J.
- **Java:** Most of the systems used are built on top of java engines so a Java distribution needs to be installed in the system before anything else.
 - On Ubuntu machines a simple list of commands is enough to install the latest distribution of Java:
 - `sudo add-apt-repository ppa:webupd8team/java`
 - `sudo apt-get update`
 - `sudo apt-get install -y oracle-java8-installer`
 - `sudo apt-get update`
 - On Windows machines we have to download the appropriate installer from <https://java.com/en/download> and execute it
- **Node.js:** Before installing Node-Red, a Node.js installation is required. We have installed Node.js version v8.9.3.
 - On Ubuntu machines we have to run the following commands:
 - `sudo apt-get update`
 - `sudo apt-get upgrade`
 - `sudo apt-get install node.js -y`
 - `sudo apt-get install npm -y`
 - On Windows machines we can download the appropriate installer from <https://nodejs.org/en/download> and execute it.
- **Node-Red:** Node-RED is a powerful visual tool for wiring together hardware devices, APIs and web-services, create flows and connect distributed components into a common IoT application [<https://nodered.org/>].
 - Installing Node-Red: The easiest way to install Node-RED is to use the node package manager, npm, which comes with Node.js [<https://nodered.org/docs/getting-started/installation>]. Installing as a global module adds the command "node-red" to your system path:
 - For Ubuntu:
 - `sudo npm install -g --unsafe-perm node-red`
 - For Windows execute CMD with administrator rights and then execute:
 - `npm install -g --unsafe-perm node-red`



- Version: We have installed Node-Red v0.17.5
- Running: after installing Node-Red as a global npm package, open a terminal and run the “node-red” command. You can then access the Node-RED editor by pointing your browser at: <http://localhost:1880>
- After accessing the editor you have to left click on the menu button (three lines on the top right corner), then click on manage palette, switch to the install tab and search for the node-red-contrib-neo4j package and install it. This will add the node required by our flows ensuring the dependency.
- **Neo4j:** In the recent years, Graph Databases have been applied in many domains such as semantic web, social networking and recommendation systems. They can efficiently handle big data problems, involving complex interconnected information, capture semantics, store, retrieve and manipulate big volumes of complex data [14]. By supporting the interrelation of data, graph databases permit the fast traversals along the edges between vertices and as models optimized for processing dense, interrelated datasets allow the detection of correlations and patterns [15] [16]. A key design characteristic for a database is its compliance to ACID (atomicity, consistency, isolation, durability) properties. Neo4j Graph Database is a fully ACID transactional database [17], guaranteeing reliable transactions and data integrity. In [18] a data model was introduced for time-varying social networks represented as graphs in the Neo4j graph database, collecting data using wearable sensors. They found graph databases very well suited to their use case due to the support for the property graph data model, the persistent, transactional storage of graphs, the support for deep graph analytics via efficient many-hop traversals and the usefulness of Cypher [19] [20] declarative query language as well. As they observed Neo4j proved to perform well when querying the sample dataset, performing exploratory data analysis, and research-oriented data mining. Most graph-based methods create only an edge between nodes to describe the interaction between a user and an item. However, these methods ignore the contextual information which may be crucial in some applications. It is observed that a user under different circumstances shows different preferences, so a recommendation system should take into consideration the decision context and the available additional information such as time, weather and location [21] [22]. In order to achieve a higher throughput we have followed the clustering approach installing:
 - Version: Neo4j enterprise edition 3.2.3
 - On Ubuntu machines we must execute the following commands in order to install Neo4j:
 - `wget -O - https://debian.neo4j.org/neotechnology.gpg.key | sudo apt-key add -`
 - `echo 'deb http://debian.neo4j.org/repo stable/' >/tmp/neo4j.list`
 - `sudo mv /tmp/neo4j.list /etc/apt/sources.list.d`
 - `sudo apt-get update`
 - `sudo apt-get install neo4j -y`
 - `sudo service neo4j restart`
 - On Windows we can follow the guide found at <https://neo4j.com/docs/operations-manual/current/installation/windows>
- **HAProxy:** In the Neo4j High Availability architecture, the cluster is typically fronted by a load balancer. The implemented cluster comprised of one server node acting as master node and 5 slave virtual machine nodes. (Figure 27)
 - Running:
 - `sudo service haproxy {start|stop|reload|restart|status}`



- config file located in /etc/haproxy
- Version: 1.8
- Operating System: Ubuntu 16.04 LTS
- www.haproxy.org

> General process information

pid = 29937 (process #1, nproc = 1)
 uptime = 2s 19s10m3s
 system limits: memmax = unlimited; ulimit-u = 4032
 maxsock = 4032; maxconn = 2000; maxpipes = 0
 current conn = 3; current pipe = 0/0; conn rate = 0/sec
 Running tasks: 3/7; idle = 100 %

active UP backup UP
 active UP, going down backup UP, going down
 active DOWN, going up backup DOWN, going up
 active or backup DOWN not checked
 active or backup DOWN for maintenance (MAINT)
 active or backup SOFT STOPPED for maintenance
 Note: "NGU/B/DRAIN" = UP with load-balancing disabled

Display option: Scope: Hide DOWN servers Hide DOWN status Hide DOWN errors Hide DOWN warnings

External resources: Primary site Updates (v1.5) Online manual

httpd		Session rate		Sessions		Bytes		Denied		Errors		Warnings		Status		Server	
Queue	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Cur	Max
Frontend	0	1,000	-	0	1,000	2,000	1,075	-	-	2,041,900	774,019	0	0	0	0	0	0
Backend																	
Queue	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Cur	Max
s1	0	0	0	0	250	0	242	2000	285	285	46m7s	512,904	198,475	0	0	0	0
s2	0	0	0	0	250	0	229	2000	285	285	46m7s	513,909	198,422	0	0	0	0
s3	0	0	0	0	250	0	236	2000	283	283	46m38s	507,600	188,000	0	0	0	33
s4	0	0	0	0	250	0	236	2000	283	283	46m37s	507,600	188,000	0	0	0	33
Backend	0	0	0	0	1,000	0	943	200	1,070	1,136	46m7s	2,041,813	772,897	0	0	0	66
2d19h UP																	

admin		Session rate		Sessions		Bytes		Denied		Errors		Warnings		Status		Server	
Queue	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Cur	Max
Frontend	0	5	-	3	3	2,000	461	-	-	21,609	120,701	0	0	7	450	0	0
Backend	0	0	0	0	5	0	1	200	451	0	0s	21,609	120,701	0	0	0	0
2d19h UP																	

FIGURE 27: HAProxy distributes requests across multiple Neo4j servers aiming to optimize resource use, maximize throughput, minimize response time and avoid overload of any single resource

- Front End: We have developed a web front end, useful for end users of our application. It provides a Graphical User Interface
 - Based on HTML, Javascript, Vue Javascript framework and other libraries
 - Running: it is deployed on our server (and cloud servers as well) and accessible in <http://snf-755174.vm.oceanos.grnet.gr>
 - Source code available (see below)
 - In order to run it locally you have to:
 - Copy index.html, functions.js and style.css in the same folder
 - Edit functions.js switching the server address to the IP that runs the node-red service
 - Run the index.html file with any browser (tested on Chrome and Firefox)
- Okeanos: We have deployed our Node-Red and Neo4j services to Okeanos cloud service for Greek Research and Academic Community
 - <https://oceanos.grnet.gr/home/>

4.1.3 User Manual

We have populated our database using open data and user-generated content. Regarding open data sources we have followed three approaches:

- 1) We integrated Neo4j in Node-Red flows and used predefined nodes provided by organizations and sites for weather in many regions.
- 2) We developed Cypher queries to access files of data and insert them in the database.
- 3) We have collected data from sensors using MQTT protocol.

Open Data sources:

- Open Weather Map
 - openweathermap.org

- It offers a special node, which regularly sends queries to the weather database of openweathermap.org and collects weather data for a specific region
- Temperature, Humidity, Pressure, visibility, wind speed, wind direction, sky cloud coverage

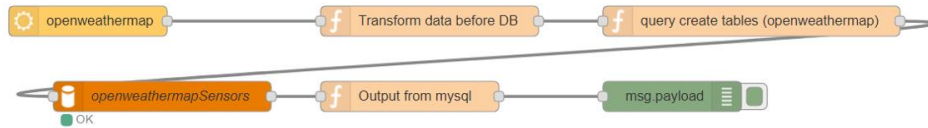


FIGURE 28: AN EXAMPLE OF TEMPLAT

- Weather Underground
 - www.wunderground.com
 - It offers a special node, which regularly sends queries to the weather database of wunderground.com and collects weather data for a specific region
 - Temperature, Humidity, Pressure, visibility, wind speed, wind direction, sky cloud coverage, solar, UV
 -
- darksky.net
 - Similar functionality to the previous two providers
- <http://www.londonair.org.uk/LondonAir/Default.aspx>

Regarding user-generated content we have used Node-Red flows. In order to effectively transmit data that are being exchanged between the sensors as well as the main system, the MQTT protocol is utilized (<http://mqtt.org/>). MQTT is a lightweight publish and subscribe messaging protocol for use on top of the TCP/IP protocol, and is ideal for use with low power sensors with limited resources such as the sensors in the solution we are implementing. MQTT is based on the principle of publishing messages and subscribing to topics. The devices transmit data in the MQTT broker using a JSON format. An example of a JSON message, its format and included information is presented below:

```
{"estate":"Dalehead","hid":"cbegt3WqEPlM","ts":1483729805,"heatmeter":{"returnTemp":"20.5",
"flowRate":"297","flowTemp":"75.7","ts":"1483729805","instant":"9280","cumulative":"30503
"}}
```

Two kinds of devices are being used in this implementation, sensors and heat-meters. The information includes the unique identity of the house which transmits the data and information about the type of the device, which can be either capturing information about the external temperature or humidity. The message also includes the actual numerical values the device captures.

4.1.4 Download - Source Code

The source code of the Recommendation Service can be found in Github at

<https://github.com/RecommendationIoT>

4.1.5 Licensing information

Since ICCS/NTUA is a non-profit Academic Research Body, we will be releasing all related BigClouT results as open source contributions under Open Source licenses. Concretely, permissive licenses, as are not restrictive licenses and it can be used to create a proprietary good, allowing a commercial exploitation and ensuring high impact. Examples of those are: Apache, BSD, etc.

5 Visualization

5.1 KNOWAGE

5.1.1 Demonstration

This section illustrates the output of the analysis defined for the scenario described in section 3.1.3; as reported in the same section, this scenario is not intended to be the final one and it might be susceptible to changes.

5.1.1.1 Demonstration – Cockpit

The cockpit presented in this section is built considering information about Fujisawa's city road damages analysed as described in section 3.1.3.2. The output of the analysis is depicted in Figure 29.

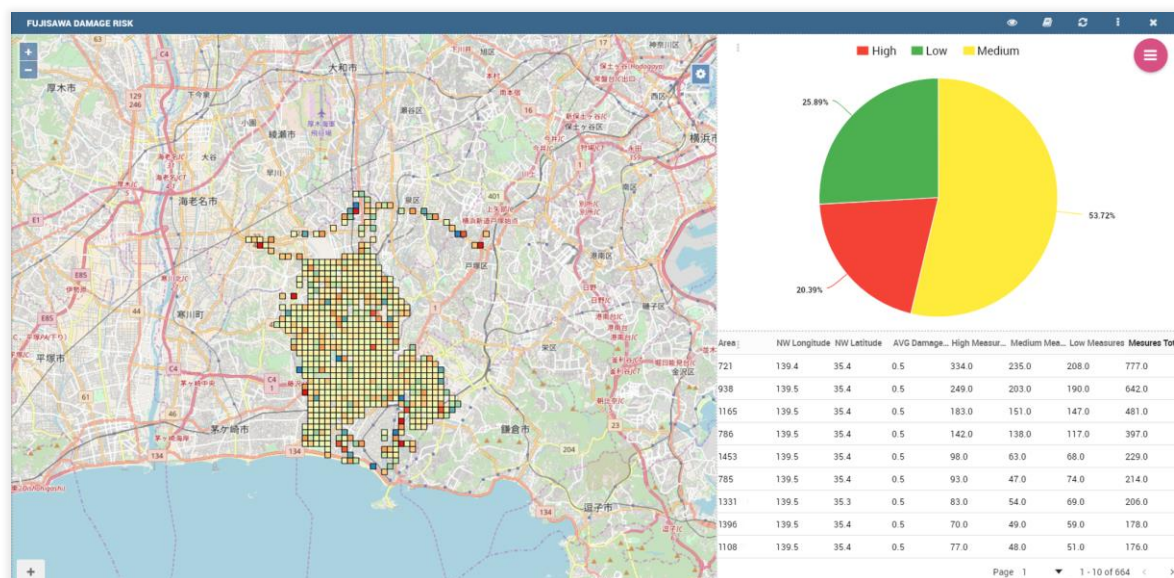


FIGURE 29: FUJISAWA DEMONSTRATOR COCKPIT

The map widget reported in Figure 29 underlines the most critical areas in the city. By clicking on a feature, the user can access additional details (Figure 30), such as the average "Damage Probability" of the selected area.

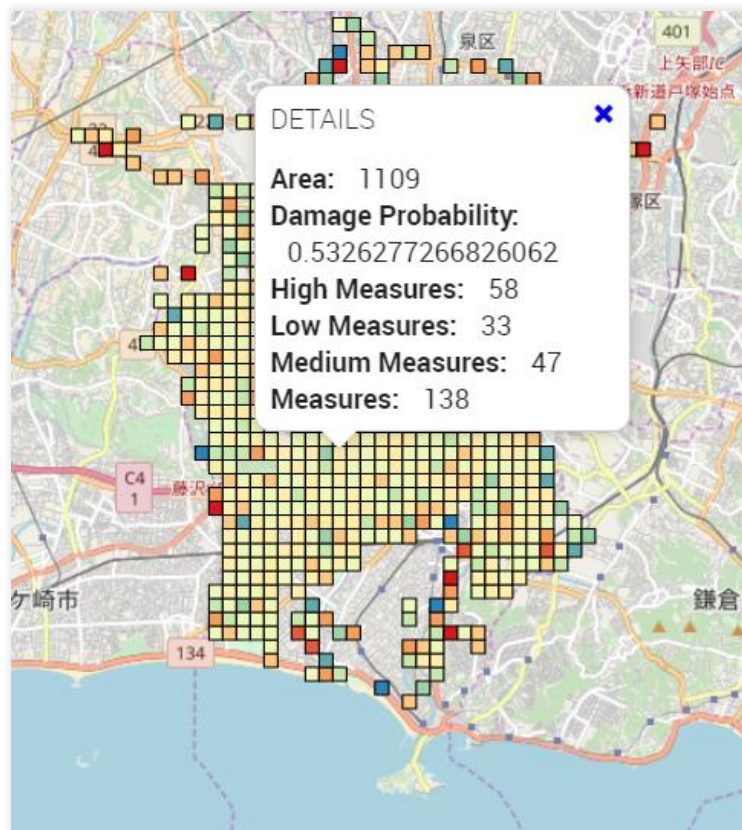


FIGURE 30: FUJISAWA COCKPIT - FEATURE DETAIL

The pie chart (reported in Figure 31) depicts the percentage of the areas classified according to three categories described in section 3.1.3.2: Low Risk (green), Medium Risk (yellow) and High Risk (red) areas.

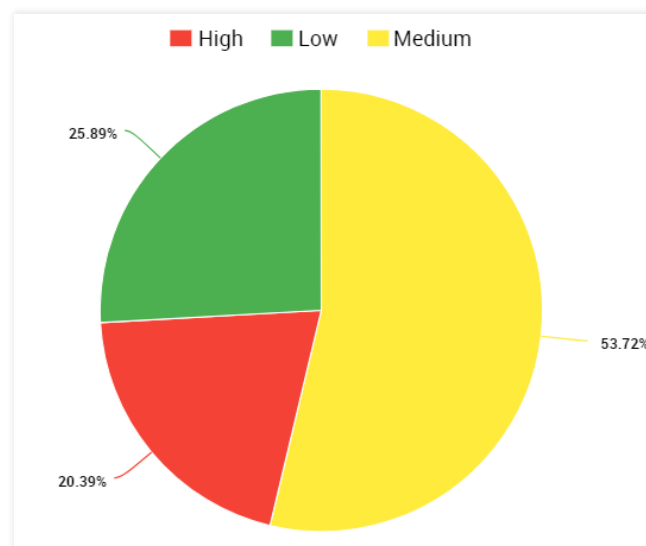


FIGURE 31: FUJISAWA COCKPIT - PIE CHART

Finally, the table widget reports the ordered list of the most critical city areas, providing information about the number of measurements collected in each area, for each of the three categories: High, Medium and Low.

Area	NW Longitude	NW Latitude	AVG Damage...	High Measur...	Medium Mea...	Low Measures	Mesures Tot
721	139.4	35.4	0.5	334.0	235.0	208.0	777.0
938	139.5	35.4	0.5	249.0	203.0	190.0	642.0
1165	139.5	35.4	0.5	183.0	151.0	147.0	481.0
786	139.5	35.4	0.5	142.0	138.0	117.0	397.0
1453	139.5	35.4	0.5	98.0	63.0	68.0	229.0
785	139.5	35.4	0.5	93.0	47.0	74.0	214.0
1331	139.5	35.3	0.5	83.0	54.0	69.0	206.0
1396	139.5	35.4	0.5	70.0	49.0	59.0	178.0
1108	139.5	35.4	0.5	77.0	48.0	51.0	176.0

Page 1 1 - 10 of 664

FIGURE 32: FUJISAWA COCKPIT - TABLE

In order to export the results, the user can click on the menu button on the top right corner of the cockpit (Figure 33), in order to export the information.

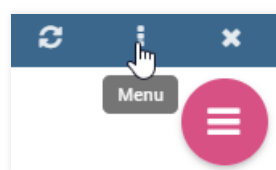


FIGURE 33: FUJISAWA COCKPIT - EXPORT

5.1.2 Package information and Installation instructions

As already described in section 3.1.2, KNOWAGE is available in two versions: the Community Edition (KNOWAGE CE) and the Enterprise Edition (KNOWAGE EE); KNOWAGE CE is freely available, whereas KNOWAGE EE is available only by subscription. This section, as well as section 3.1, refers to the KNOWAGE CE version of the suite.

In the context of visualization, it is important to underline the differences between the two versions of the tool. In order to create charts or cockpit, the main framework used by KNOWAGE is Highcharts JS [23]. This framework is known to be freely available with restrictions, for instance it is free for personal websites, school sites or non-profit organization sites. Other organizational sites need a subscription to use Highcharts' features. The subscription to KNOWAGE EE includes the commercial license of Highcharts JS. The Community Edition version gives the chance to the user to choose which visualization framework to be used. Indeed, during the installation of the tool, through the installation wizard, the user can choose to install Chart.js [24] or Highchart JS, as depicted in Figure 34.

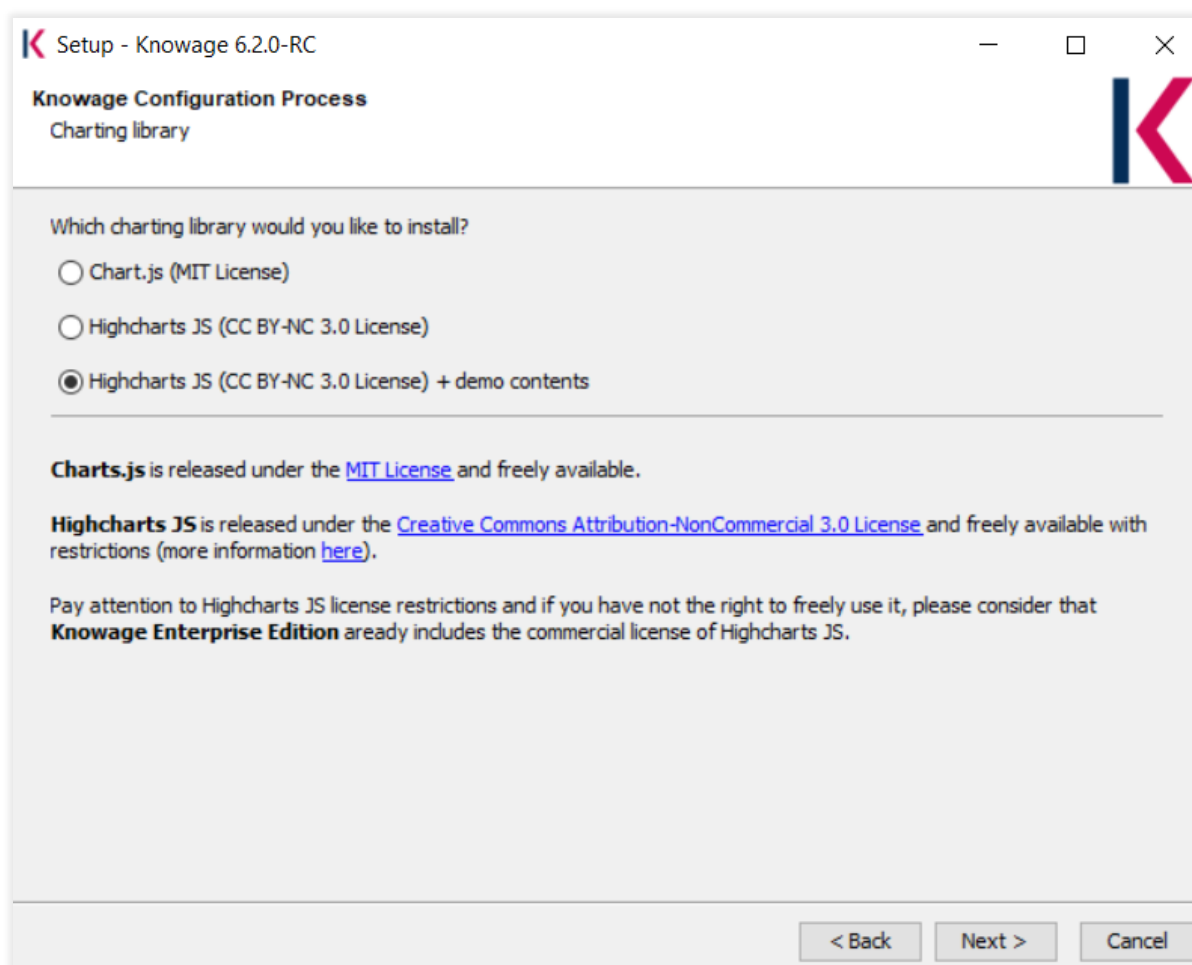


FIGURE 34: KNOWAGE CE VISUALIZATION FRAMEWORK

By selecting Highchart JS, with or without the demo contents, the user would need to take care about its license restrictions.

5.1.3 User Manual

This section of the document covers the basic steps needed to visualize data through the definition of a Cockpit, detailed information can be found in [11]. A Cockpit is a multi-sheet and interactive document where the user can build complex visualization by creating, configuring and customizing several widgets. KNOWAGE provides the following widgets:

- **Text:** through this widget a user can add text information to the cockpit.
- **Image:** which is used to add images to the cockpit.
- **Table:** which creates a table in the cockpit.
- **Cross Table:** which is used to create cross or pivot tables.
- **Html:** with this widget the user can add custom html to the cockpit.
- **Document:** which is used to add previously created documents to the cockpit.
- **Chart:** which is used to add several charts type into the document.
- **Map:** with this widget dataset containing spatial attribute can be easily displayed in a map. Map widget is currently in an early stage and it will be released in next versions of KNOWAGE CE [10].

- **Selector:** which helps creating filters using directly the fields of the used datasets. Using this approach, the user doesn't have to define Analytical Drivers.
- **Active Selections:** which shows the current active selections on the cockpit.

This user guide is intended to guide the user in the creation of a Cockpit document and the configuration of:

- Chart widget
- Map widget

Cockpit Document

After a successful login, the user will access KNOWAGE's home page. In order to create a Cockpit document, the user should open the main menu and click on the *Workspace* link in the left panel, as illustrated in Figure 35.

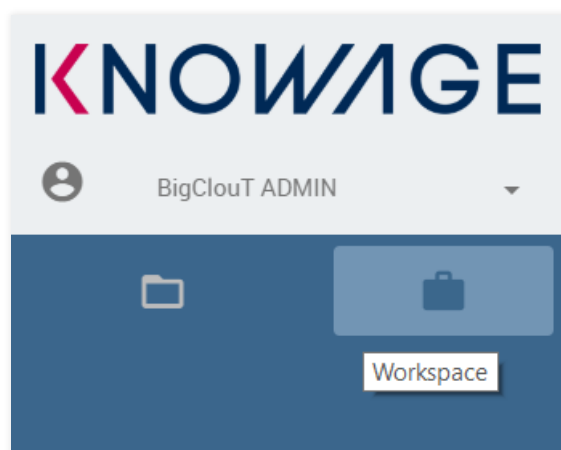


FIGURE 35: KNOWAGE WORKSPACE LINK

The user should then click on the *Analysis* section and click on the plus button in the top right corner of the page and select *Cockpit*. The cockpit interface is an empty page with a toolbar containing different options as depicted in Figure 36.



FIGURE 36: KNOWAGE COCKPIT TOOLBAR

Starting from left to right, these buttons are used to:

- **Clear cache:** which cleans the temporary data stored for the cockpit.
- **Selections:** which displays the active selections of the cockpit.
- **Data configuration:** which opens a window where the user can manage datasets, the association between datasets and the refresh frequency.
- **General configuration:** which opens a window where the user can set the general cockpit options.
- **Add widget:** which opens a window where the user can select the widget to be added to the cockpit, as depicted in Figure 37.

- **Save as:** which is used to save the cockpit document.
- **Cockpit menu:** which opens or closes the toolbar.

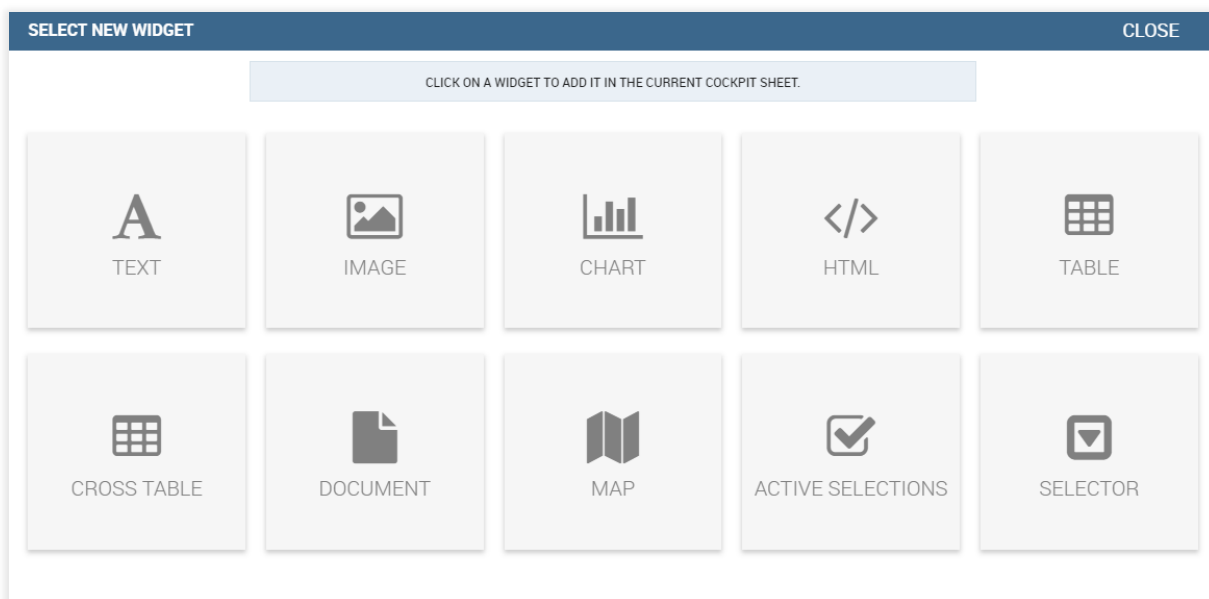


FIGURE 37: KNOWAGE WIDGET PANEL

The main operations, the user should perform, are related to the management of one or more datasets and the definition of widgets which are used to display datasets information. Once the Cockpit is completed the user should save the document inserting the Cockpit name.

Chart Widget

By selecting the chart widget, the chart widget configuration window, depicted in Figure 38, will appear.

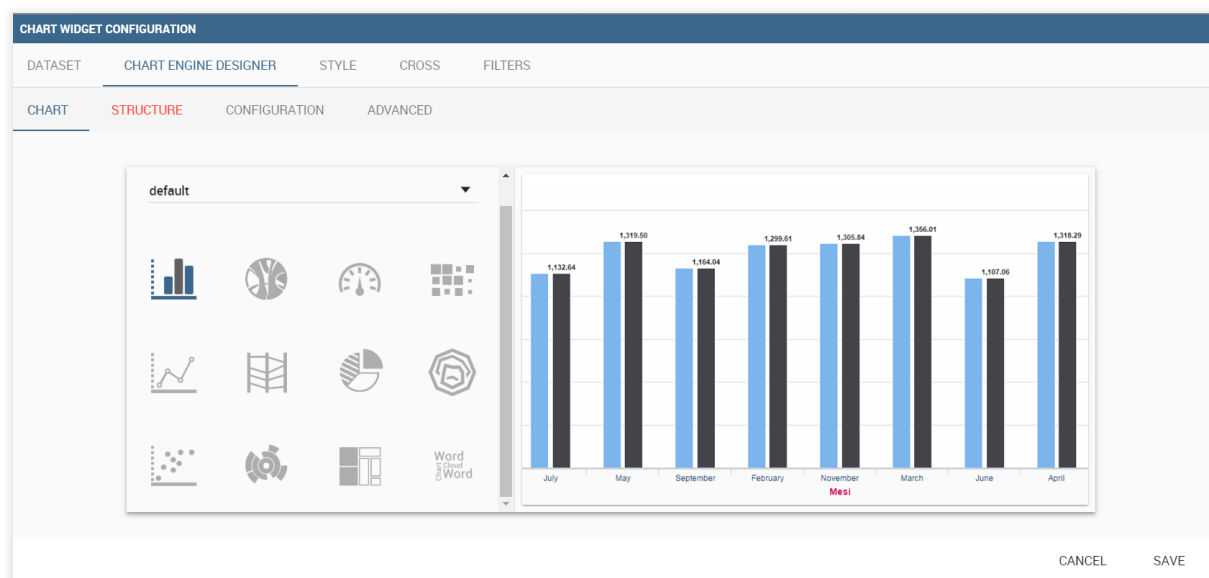


FIGURE 38: KNOWAGE CHART WIDGET WINDOW



The first step to accomplish is the selection of a Dataset from the appropriate section of the window. The *Chart Engine Designer* section is used for selecting the specific chart the user wants to create, thanks to its *Chart* inner section. KNOWAGE provides several charts, in particular: Bar, Chord, Gauge, Heatmap, Line, Parallel, Pie, Radar, Scatter, Sunburst, Treemap and Wordcloud. Every chart can be customized following the specific needs of the analysis. The *Structure* inner section is used to select which field of the dataset the chart should display, selecting one or more attributes to be used as categories and one or more measurements to be used as data series, as showed in Figure 39.

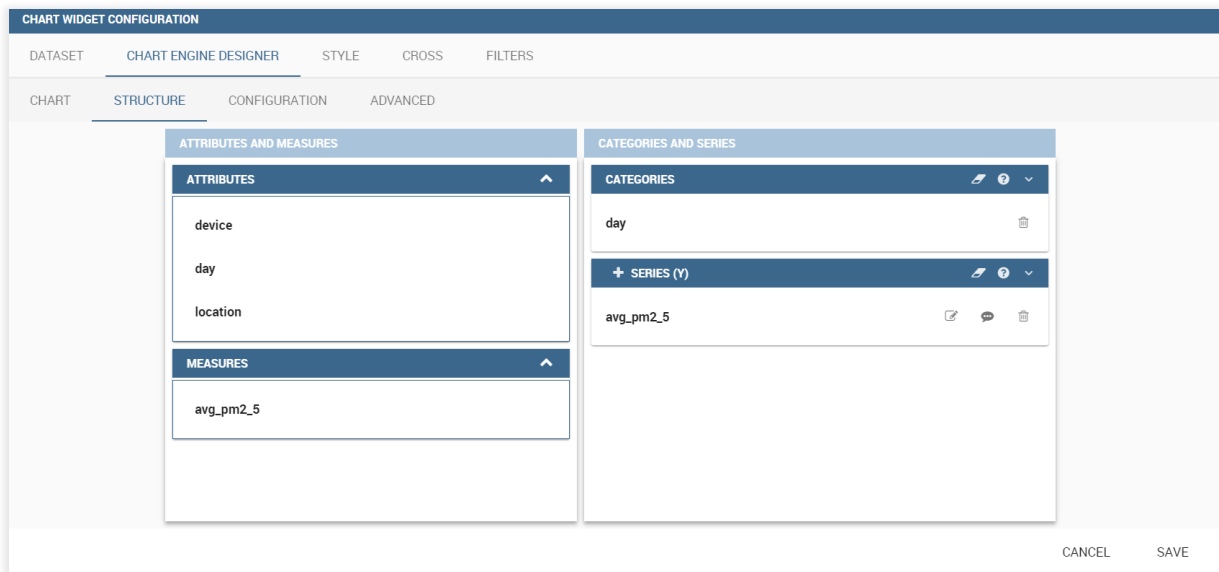


FIGURE 39: KNOWAGE CHART STRUCTURE

The *Configuration* inner section let the user manage several properties of the chart such as the title, the font, the legend and the “no-data message”, as illustrated in Figure 40.

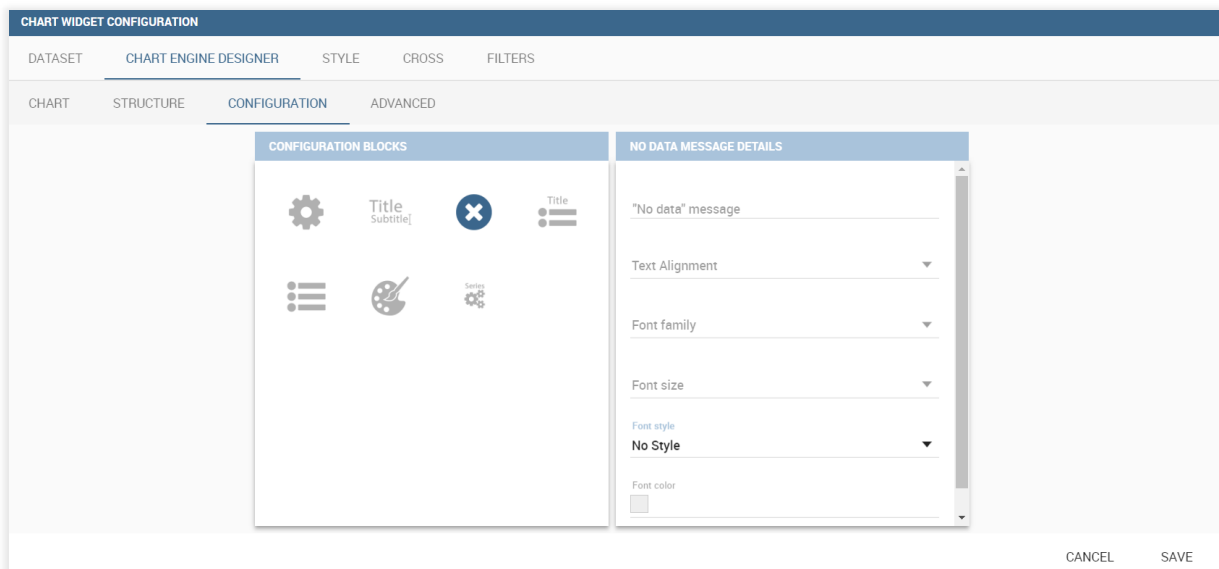


FIGURE 40: KNOWAGE CHART CONFIGURATION

The *Advanced* inner section shows a JSON representation of the configured chart, Figure 41.

Type	Name	Label	Target
DATASET	GarbageTruck_AvgPm2.5 per hour	GarbageTruck_AvgPm25_H	<input type="checkbox"/>

Column	Coordinates Type	Coordinates Format	Json Feature Type
location	string	lon lat	

Column	Alias	Type	Aggregation function	Show on detail	Show on map
avg_pm2_5	avg_pm2_5	MEASURE	SUM	<input type="checkbox"/>	<input type="checkbox"/>
hour	hour	ATTRIBUTE		<input type="checkbox"/>	
day	day	ATTRIBUTE		<input type="checkbox"/>	
device	device	ATTRIBUTE		<input type="checkbox"/>	

FIGURE 43: KNOWAGE MAP METADATA CONFIGURATION

By clicking on the style button, the user can choose the way the measurements should to be displayed on the map, Figure 44. Several options are available for the visualization of the measurements in the map, the most common one is the simple marker but the user can use also clusters or heatmaps.

Type	Name	Label	Target
DATASET	GarbageTruck_AvgPm2.5 per hour	GarbageTruck_AvgPm25_H	<input type="checkbox"/>

VISUALIZATION TYPES

Markers

Color

FIGURE 44: KNOWAGE MAP STYLE CONFIGURATION

Additional and more detailed information about the usage of KNOWAGE can be found in [\[11\]](#)

5.1.4 Download – Source Code Repository

As previously defined in section 3.1.4, the installation packages of KNOWAGE CE (current stable version 6.1, whereas version 6.2 is release candidate) are available at [\[4\]](#), whereas its source code is available on a dedicated GitHub page [\[12\]](#).



5.1.5 Licensing information

Source code of KNOWAGE CE is released under the terms of GNU Affero GPL v3 [13] and with this version of the tool the user must be aware of the usage restrictions of Highchart JS. The subscription to KNOWAGE EE already includes the commercial license of Highcharts JS.



6 Context Management and self-awareness

6.1 “Deep on Edge” Delivery and Usage

6.1.1 *Demonstration – Demonstrator*

When the DeepOnEdge system is initialized with the specific command, it starts capturing the video image (input from the video capture device) in per-frame basis.

DeepOnEdge opens a window on the screen, illustrated as follows. Frame-by-frame basis, the result of road marking damage detection will be displayed.

In case that any damage is detected, as show in Figure 45, DeepOnEdge shows the result “damage” with detection confidence (e.g., 56.8% in this figure).

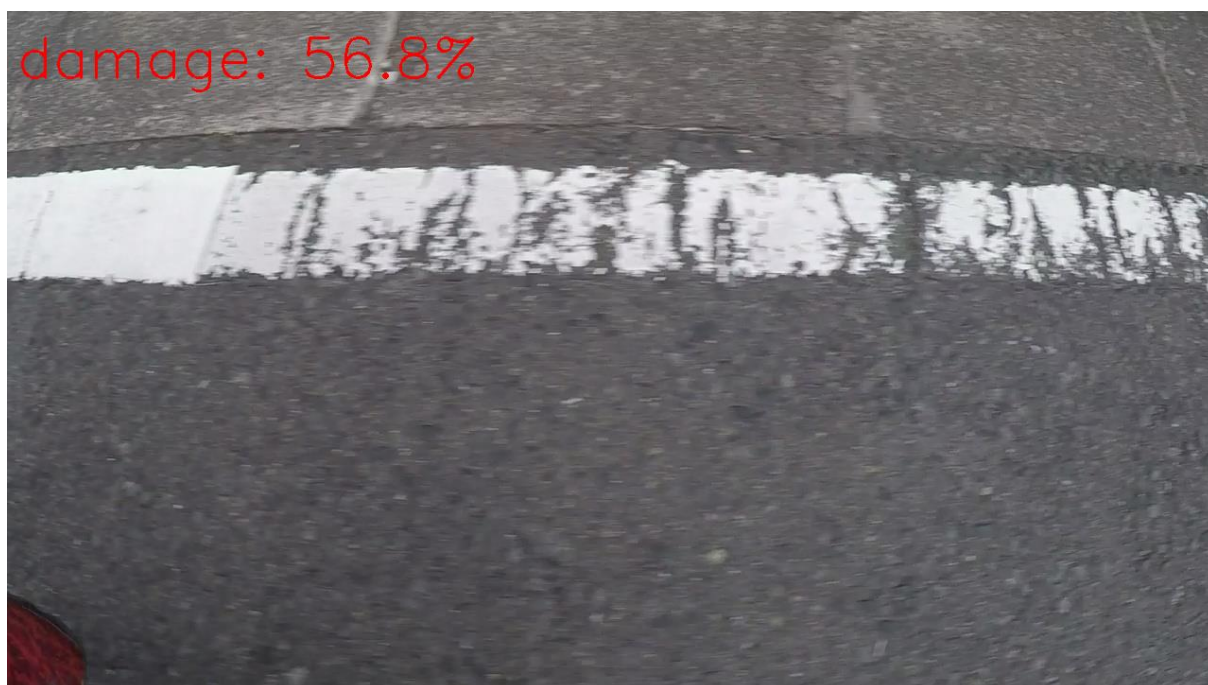


FIGURE 45: THE CAPTURED ROAD IMAGE WITH DETECTED DAMAGE ON THE ROAD MARKINGS.

In case that any damage is not detected, as show in Figure 46, DeepOnEdge shows the result “no damage” with detection confidence (e.g., 90.3% in this figure).

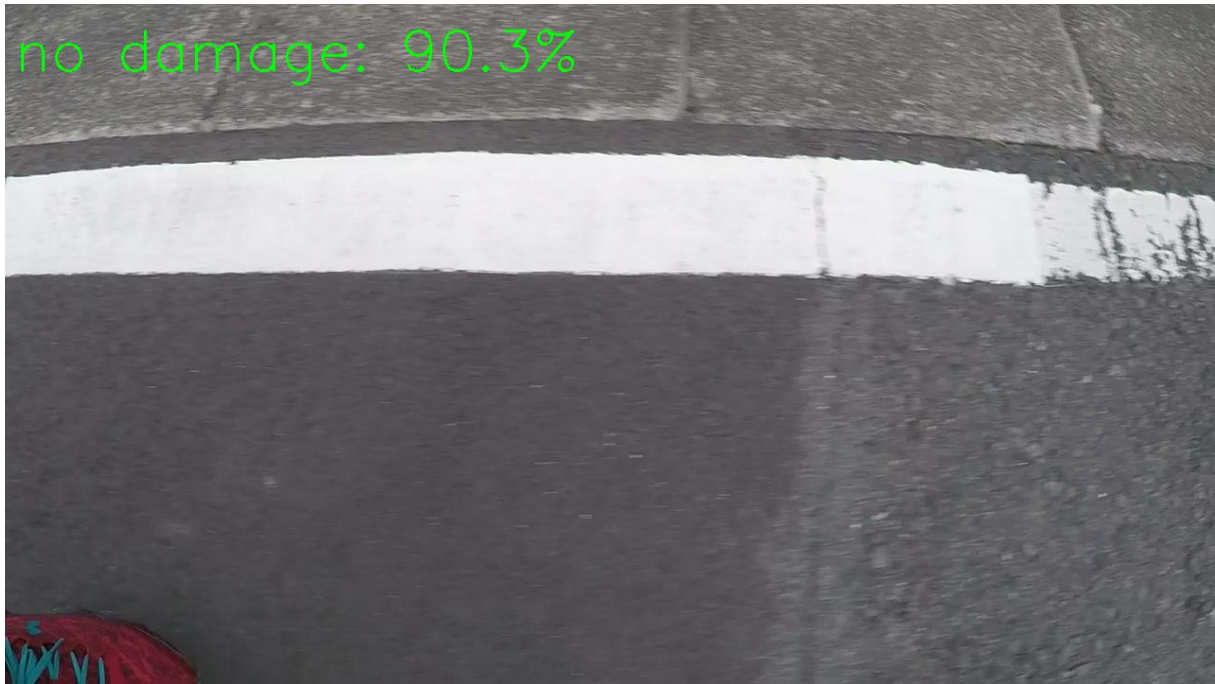


FIGURE 46: THE CAPTURED ROAD IMAGE WITHOUT ANY DETECTED DAMAGE ON THE ROAD MARKINGS.

6.1.2 Package information and Installation instructions

Package information and installation instructions.

- (OS) DeepOnEdge does not need to be installed but be placed in a directory with the right permission to be executed. What has to be paid attention to is using the appropriate OS: e.g. Ubuntu, CentOS or other Linux and MacOS are supported.
- (python) The basic prerequisite for DeepOnEdge is python3. More specifically, the python version used is 3.5 or 3.6.
- (Required modules) On the operating system, DeepOnEdge needs the following modules:
 - pytorch version ≥ 0.4
 - opencv version ≥ 3.2
 - numpy ≥ 1.11

These modules can be installed by pip or conda install.

6.1.3 User Manual

Setting-up / Initialisation / Instantiation instructions.

- Set-up
 - Make the attached camera recognized by DeepOnEdge.
- Execution
 - From the command line, start DeepOnEdge with the following command.
\$ python main.py
- Expected Input / Output.
 - Input
 - ✧ Video image capturing the road surface.
 - ✧ Specification
 - Image quality (resolution): 640 pixel x 480 pixel (or better)

- Frame speed: 10fps (or better)
- Output
 - ✧ Showing the captured video on the screen
 - ✧ Result of the real-time edge processing containing the detection result of road markings damages.
 - Caption “Damage” if a marking is detected to be damaged.
 - Caption “No Damage” if a marking is detected not to be damaged.
- Available Modes / Commands
 - None

6.1.4 Download – Source Code Repository

The source code of DeepOnEdge can be found in Github at

<https://github.com/makora9143/DeepOnEdge>

6.1.5 Licensing information

Currently, DeepOnEdge is distributed as closed source software. However, first discussions have been carried out by A Partner of the Consortium in order to release it as Open Source software.



7 Conclusions

This deliverable has presented the prototypes and demonstrators of the city data processing module in the BigClouT architecture. The role of the modules is to apply high-level analysis over city data to extract city knowledge. The demonstrations have shown that the functionalities of city data processing module can process city data and successfully extract useful knowledge from city data by way of aggregation, event detection, recommendation, machine learning, etc.

The source codes of the prototypes and demonstrations are basically provided online. Therefore, the provided functionalities can be easily reused and exploited to other cities in EU and Japan, as well as in other countries.



8 Bibliography

- [1] Rubenfire M, Das R, Gracik T, Wang L, Morishita M, Bard RL, Jackson EA, Fitzner CA, Ferri C, Brook RD. Giorgini P, "Particulate matter air pollution and ambient temperature: opposing effects on blood pressure in high-risk cardiac patients. ".
- [2] Coexistence of PM2.5 and low temperature is associated with morning hypertension in hypertensives., "Imaizumi Y, Eguchi K1, Kario K. ".
- [3] Ercegovac. V., Gemulla. R., Eltabakh. M., Kanne. C., Ozcan. F., Shekita. Beye. K., "Jaql: A Scripting Language for Large Scale Semi-structured Data Analysis," *Proceedings of the VLDB Endowment* 2011.
- [4] KNOWAGE CE download. [Online]. <https://www.knowage-suite.com/site/ce-download/>
- [5] Java SE Development Kit. [Online]. <http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>
- [6] MySQL Server. [Online]. <https://www.mysql.com/>
- [7] MariaDB Server. [Online]. <https://mariadb.org/>
- [8] KNOWAGE CE Installer - Getting started. [Online]. https://www.knowage-suite.com/site/wp-content/uploads/2018/03/Knowage-CE-Installer-Getting-started_6_1_1.pdf
- [9] Apache Tomcat. [Online]. <http://tomcat.apache.org/>
- [10] KNOWAGE - Product comparison. [Online]. <https://www.knowage-suite.com/site/product/product-comparison/>
- [11] KNOWAGE CE Manual. [Online]. http://download.forge.ow2.org/knowage/Knowage_6.x_CE_Manual.pdf
- [12] KNOWAGE Labs - GitHub. [Online]. <https://github.com/KnowageLabs>
- [13] GNU Affero General Public License Version 3. [Online]. <https://www.gnu.org/licenses/agpl-3.0.html>
- [14] J. J. Miller, "Graph database applications and concepts with Neo4j," *Proceedings of the Southern Association for Information Systems Conference*, vol. 2324, p. 36, 2013.
- [15] M. A. Rodriguez and P. Neubauer, "Constructions from dots and lines," *Bull. Am. Soc. Inf. Sci. Technol.*, vol. 36, no. 6, pp. 35–41, 2010.
- [16] M. A. Rodriguez and P. Neubauer, "The graph traversal pattern," *ArXiv Prepr. ArXiv10041001*, 2010.



- [17] [Online]. <https://neo4j.com/developer/cypher-query-language/>
- [18] M. Quaggiotto, A. Panisson, and A. Averbuch C. Cattuto, "Time-varying Social Networks in a Graph Database: A Neo4J Use Case," *First International Workshop on Graph Data Management Experiences and Systems*, pp. 11:1–11:6, 2013.
- [19] [Online]. <https://neo4j.com/developer/cypher-query-language/>
- [20] J. Webber, and E. Eifrem. I. Robinson, "Graph databases: new opportunities for connected data.," in *O'Reilly Media, Inc.*, 2015.
- [21] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions," vol. 17, no. 6, p. 2005.
- [22] J. He, G. Huang, and Y. Zhang W. Yao, "SoRank: Incorporating Social Information into Learning to Rank Models for Recommendation," *Proceedings of the 23rd International Conference on World Wide Web*, pp. 409–410, 2014.
- [23] Highcharts. [Online]. <https://www.highcharts.com/>
- [24] Chart.js. [Online]. <https://www.chartjs.org/>
- [25] Rubenfire M, Das R, Gracik T, Wang L, Morishita M, Bard RL, Jackson EA, Fitzner CA, Ferri C, Brook RD. Giorgini P, *Particulate matter air pollution and ambient temperature: opposing effects on blood pressure in high-risk cardiac patients.*
- [26] Rubenfire M, Das R, Gracik T, Wang L, Morishita M, Bard RL, Jackson EA, Fitzner CA, Ferri C, Brook RD. Giorgini P, *Particulate matter air pollution and ambient temperature: opposing effects on blood pressure in high-risk cardiac patients..*
- [27] Rubenfire M, Das R, Gracik T, Wang L, Morishita M, Bard RL, Jackson EA, Fitzner CA, Ferri C, Brook RD. Giorgini P, "Particulate matter air pollution and ambient temperature: opposing effects on blood pressure in high-risk cardiac patients. ".

