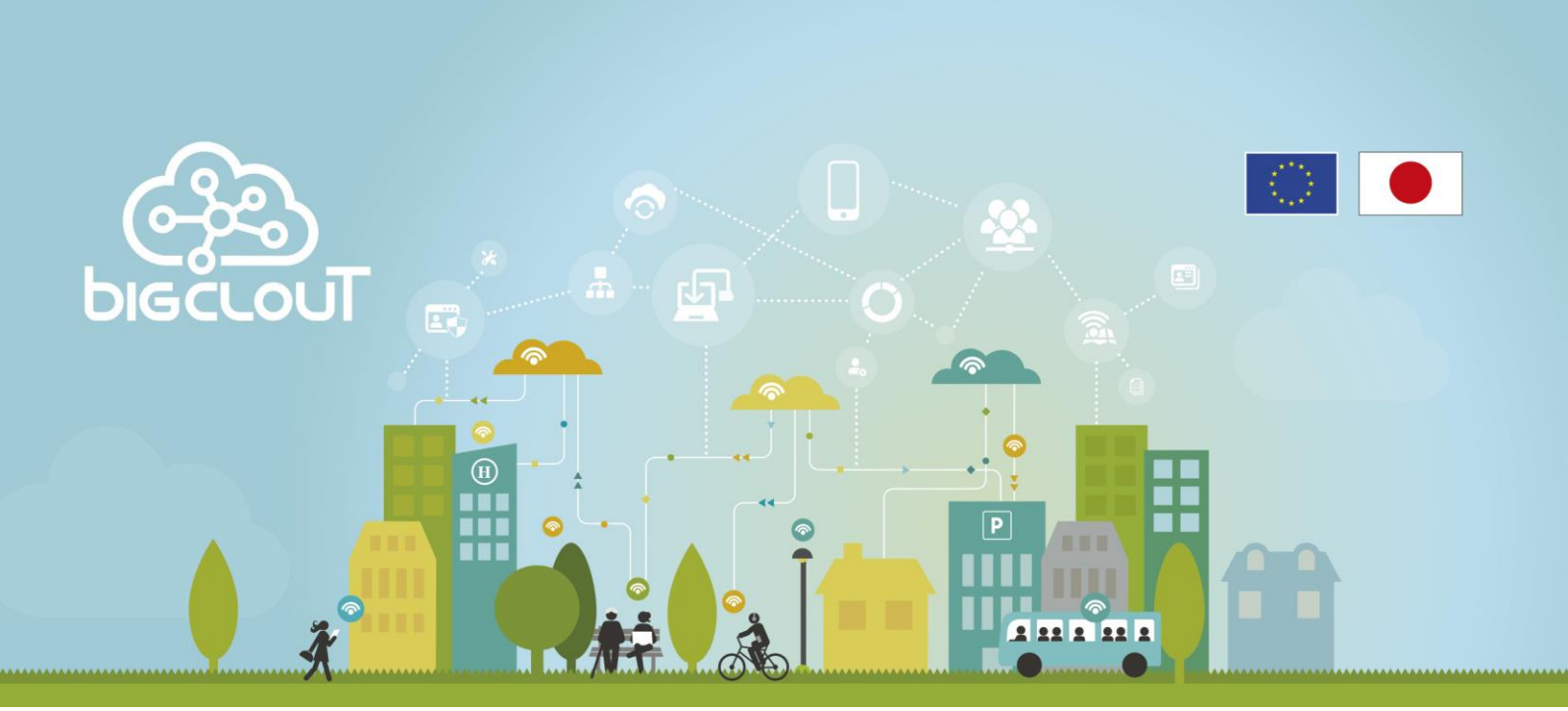


WP2 Programmable Smart City
D2.4 Distributed data-flow programming tool -Demonstration

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: **LANC** JP Editor**: KEIO** Nature: Demonstrator

Dissemination: PU

Contractual delivery date: 2018-07-01

Submission Date: 2018-07-04

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

ABSTRACT

This deliverable describes the first demonstrators of the Distributed data flow programming tool used as part of the BigClouT architecture.

Disclaimer

Revision history

Revision	Date	Description	Author (Organisation)
V0.0	2018-4-27	Initial ToC	RJL (LANC)
V0.1	2018-5-05	Initial text	RJL (LANC)
V0.2	2018-5-21	More text	RJL (LANC)
V0.3	2018-5-23	Tidied + added diags	RJL (LANC)
V0.4	2018-5-28	Keio text	TY (KEIO)
V0.5	2018-5-30	Lanc scenario	MB (LANC)
V0.6	2018-06-04	Merged updates	RJL (LANC)
V0.7	2018-06-08	Finalized for review	RJL/TY/MB
V1.0	2018-06-18	Reviewers comments	RJL

TABLE OF CONTENT

LIST OF FIGURES

1 INTRODUCTION

1.1 Scope of the Document

This deliverable describes the demonstration of the Distributed Dataflow programming tool developed within WP2 of the BigClouT project and designed to offer an easy programming tool for the development of edge based Smart City applications.

The tool, referred to as Distributed-NodeRed (D-NR), is based on the open source visual programming tool, Node-RED, augmented with support for distributed and edge processing and integrated into the BigClouT architecture.

1.2 Target Audience

The target audience of this deliverable are mainly the following groups:

- BigClouT project members / developers – who are currently involved in the project and are responsible for this platform or other parts of BigClouT. This document can serve as a reference to facilitate members to understand the existing functionality, further implement new ones, or modify existing ones.
- Smart City Ecosystem integrators - who plan to develop a smart city ecosystem based on the BigClouT reference. The document outlines the relevant components of the tool and details the technical implementation, which is a good source of information to provide an in-depth understanding of the platform.

1.3 Structure of the Document

The document provides a general overview of the goals of the tool (section 2), and in depth discussion of the architecture and features (section 3) and then describes (section 4) the core demonstrator that has been developed to showcase the capabilities of the tool when used as part of an ongoing trial – in this case in Fujisawa. Finally, section 5 discusses 2 support demonstrators that have also been developed that showcase specific aspects of the tool.

2 GOALS OF THE DISTRIBUTED DATA FLOW PROGRAMMING TOOL: DISTRIBUTED NODE-RED

The distributed data-flow programming tool is designed to address a number of key issues:

- conform to the BigClouT architecture as a service composition tool (see D1.3)
- provide an easy to use visual programming metaphor incorporating data flow
- support the BigClouT edge processing needs

To achieve these aims, the project has extended the open source tool, Node-RED, which provides a basic data flow capability combined with a visual programming model that supports JavaScript. Extensions to the open source Node-RED language include support for the distributed edge processing based on a 'fog computing' model as well as integration into the overall BigClouT architecture.

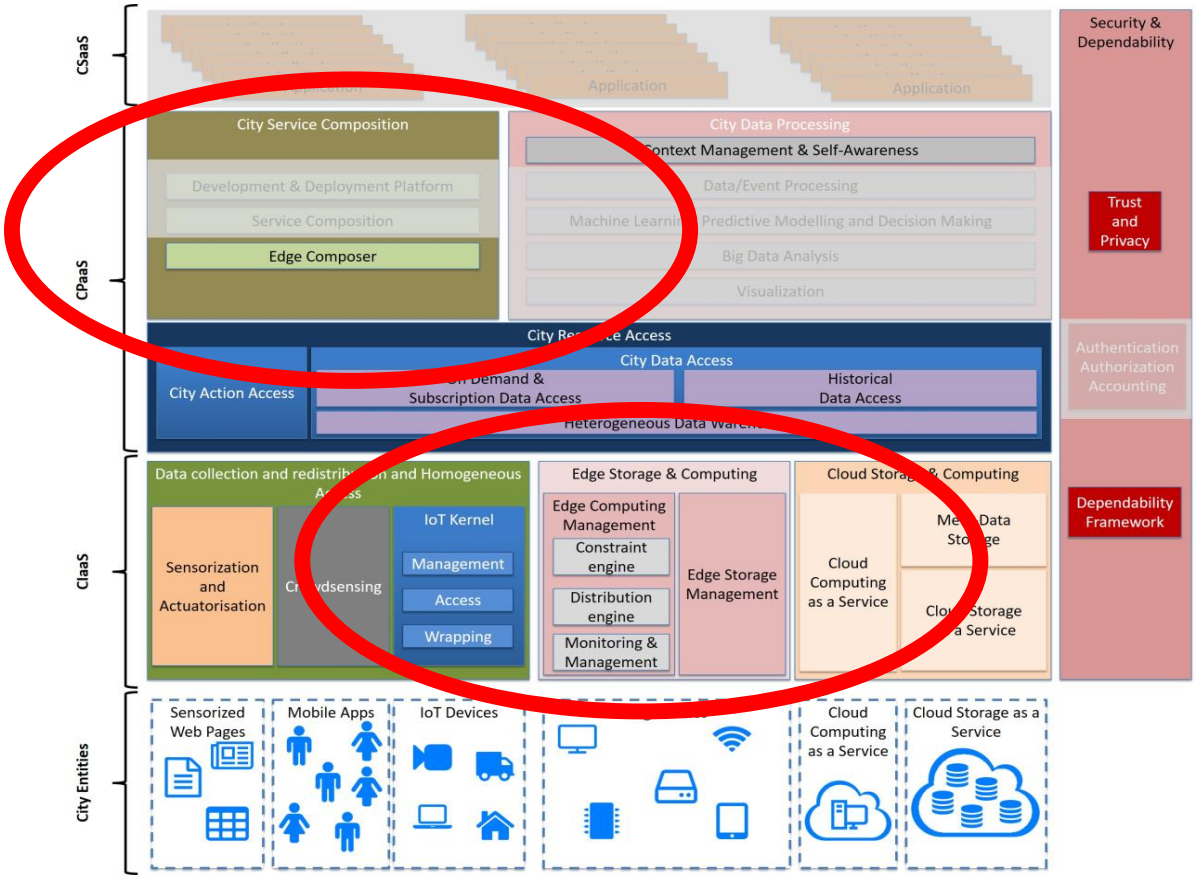


FIGURE 1 : DISTRIBUTED DATA FLOW PROGRAMMING TOOL - ARCHITECTURAL POSITIONING

These extensions are implemented into the core D-NR platform (which supports the edge processing mechanisms) and to the visual programming tool (D-NR studio) that supports the specification and application of the policy developers wish used to control the distributed nature of the BigClouT application. The core D-NR platform forms part of the Edge storage, a computing subsystem of the BigClouT architecture (specifically the edge computing management system (See Figure 1 above) and the D-NR studio implements the edge composer tool which is part of the City Service Composition subsystem.

2.1 Overview of the demo

To demonstrate the core capabilities of BigClouT's distributed data flow programming tool we have used the core tool to implement a distributed processing feature that is part of the Fujisawa core infrastructure field trial. In this core infrastructure trial, Fujisawa city staff are able to monitor the status of the road network, and in particular the road surface conditions and the quality of the road markings, by analysing real time video captured by a fleet of garbage trucks as they work within the city. The demonstrator shows how data is captured and analysed by edge processors running in the garbage trucks, data is collated from the city using D-NR, and then issues are analysed by viewing specific video recordings. A specific feature of the demo, apart from its use of WP2 T2.4 technologies, i.e. distributed edge processing and the distributed data flow programming tool, is its ability to protect privacy by hiding data such as faces and car number plates using a WP3 contribution “Deep on Edge”. Lastly, another WP3 tool, Knowage, is used to visualise aggregated data from the city as a whole.

In addition to the Fujisawa trial demo we have developed two other demonstrators that highlight key aspects of the D-NR tool. The first is a demonstrator aligned with the Bristol smart Energy

trial. It consists of a set of home sensors gathering energy (and pollution) data from the home using a local instance of the D-NR run-time. These data are then batched and sent via internet to a cloud server also running an instance of the D-NR run-time. The data are then analysed and presented using the visualisation capabilities of D-NR.

A third demo is also available that shows the internal workings of the D-NR tool and its dynamic capabilities. The demo shows the use of edge processing to distribute load across a distributed application as network and computation resources change due to processing needs. In the demo, a smart traffic application is shown which monitors traffic in a city. As traffic builds towards rush hour, more computation and network resources are required to monitor the traffic and the D-NT *dynamically adapts* to meet the need.

3 DISTRIBUTED NODE-RED (D-NR)

The distributed data flow programming tool D-NR is split into two core components: Studio and Platform. Studio provides a visual programming tool allowing service composition by dragging and dropping application components onto a visual drawing board and wiring the components together to form a flow. While the D-NR platform supports edge processing and dynamic load balancing across the BigClouT network.

3.1 Service composition: D-NR studio (visual programming tool)

Service composition uses a set of pre-defined building blocks that are wired together to form the application logic.

In Figure 2 a basic air quality application is shown that reads data from the BigClouT data warehouse (sensor data from Bristol) and visualises the data on a dashboard.

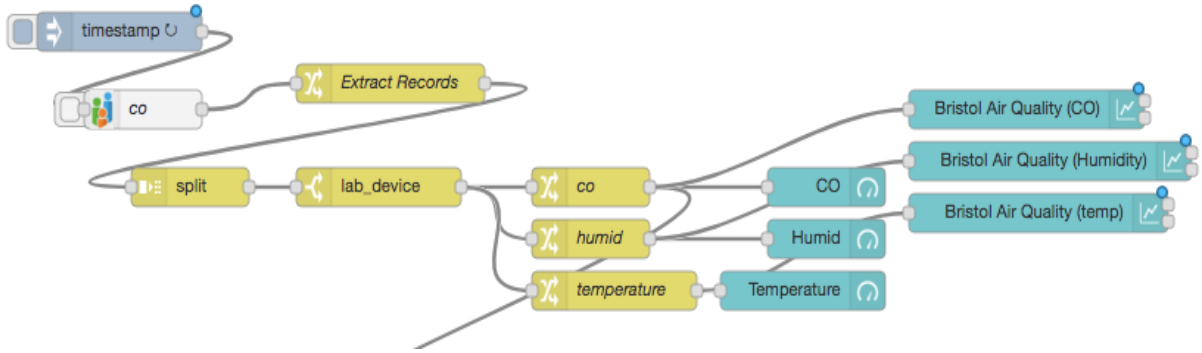


FIGURE 2: VISUAL PROGRAMMING USING A DRAG AND DROP METAPHOR

The data flow from left to right through the application, traversing the 'wires' between the processing nodes. Each node receives data, carries out some processing and sends the data to the next processing node in the flow.

3.2 Edge processing: D-NR platform (fog/edge computing capability)

To extend Node-RED to meet the BigClouT need for a service composition tool with support for edge processing we addressed a number of key issues:

- Developing a model to describe devices and their capabilities and incorporating into the Node-RED development tool.
- Supporting a language transparent mechanism to allow nodes in the application flow to be moved to remote devices.
- Defining a constraint model that allowed application developers to specify constraints for different parts of their application flow, which then drove the underlying distribution and replication mechanisms.

We introduced the notion of *device* to the dataflow language. Accordingly, every node in a dataflow program is augmented with a new *device Id* constraint that specifies on which device the node should be deployed and run. For example, a node can be constrained to be deployed on an edge device, a mobile host, a cloud server or on any intermediary device across the edge to the cloud.

The second augmentation made to Node-RED was the notion of "remote wires" or "remote arcs". Since the nodes may run on separate devices, the existing Node-RED wires - which represent dataflow links between processing nodes - have to support inter-device communication to handle the situation where a flow is broken up and its nodes are distributed to several devices. This is implemented using a publish/subscribe communication mechanism that binds the nodes together. The key idea is to leverage the node identifications as the topic for publishing and subscribing. Further, a flow transformation process is applied so that the nodes that do not meet the deployment requirement (e.g. run on "mobile", "laptop" or "server") will be replaced with a *wire in* or a *wire out* node. *wire in* nodes subscribe to the communication broker so that it can receive data from the external node running on a different device. *wire out* nodes receive data from the local node and publishes it to the communication broker so that the *wire in* node from the other side can pick it up. Figure 3 illustrates this process of supporting the distributed deployment of a Node-RED flow across multiple devices.

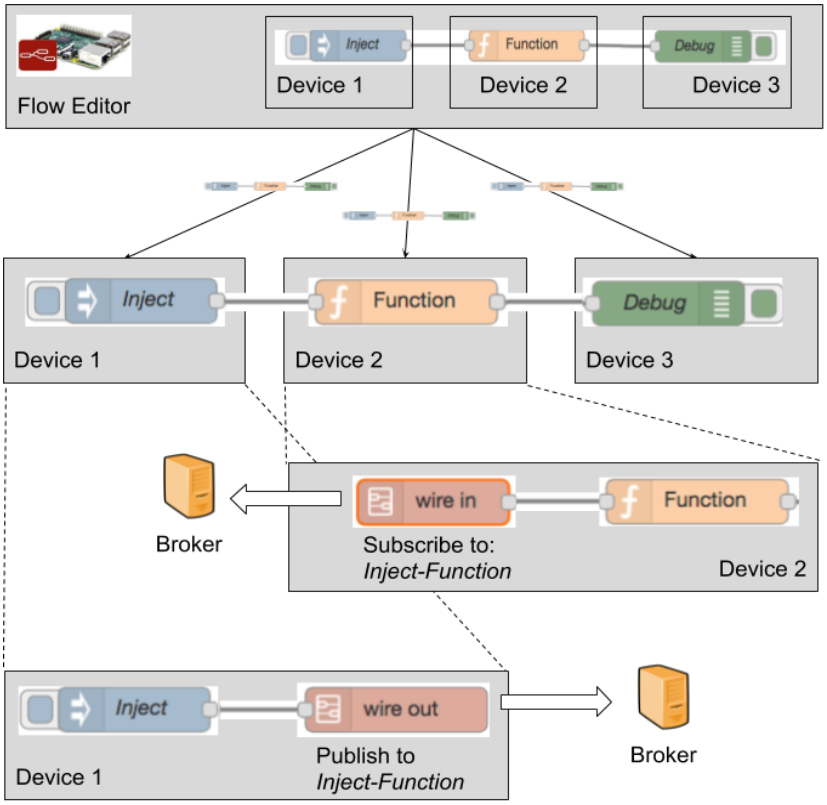


FIGURE 3: DISTRIBUTED DEPLOYMENT TO EDGE DEVICES USING D-NR IN BIGCLOUT

3.3 Constraints: Supporting More Complex, Larger Scale Applications

To support larger scale BigClouT applications, we augmented the basic device specification capability with a constraint tool. This allowed developers to define a set of constraints for a part of the flow, e.g. a node should run on a device with 4MB of memory, a 4 core CPU and is physically located in the Henleaze area of Bristol city.

3.3.1 Constraint-based Distribution

This more sophisticated mechanism allows scenarios such as:

- A sensor node mounted on a moving vehicle could be restricted to operate in a certain location.
- A vision processing node might be restricted to operate in a more capable computing device.

To address these needs, we introduced the *constraint* primitive as a broader abstraction that specifies how a node is deployed and run in a distributed computing setting. Accordingly, every node in a dataflow program is augmented with a *constraint* property that defines how the deployment is carried out. In BigClouT, a *constraint* involves the requirements on device identification, computing resources such as CPU and memory and physical location.

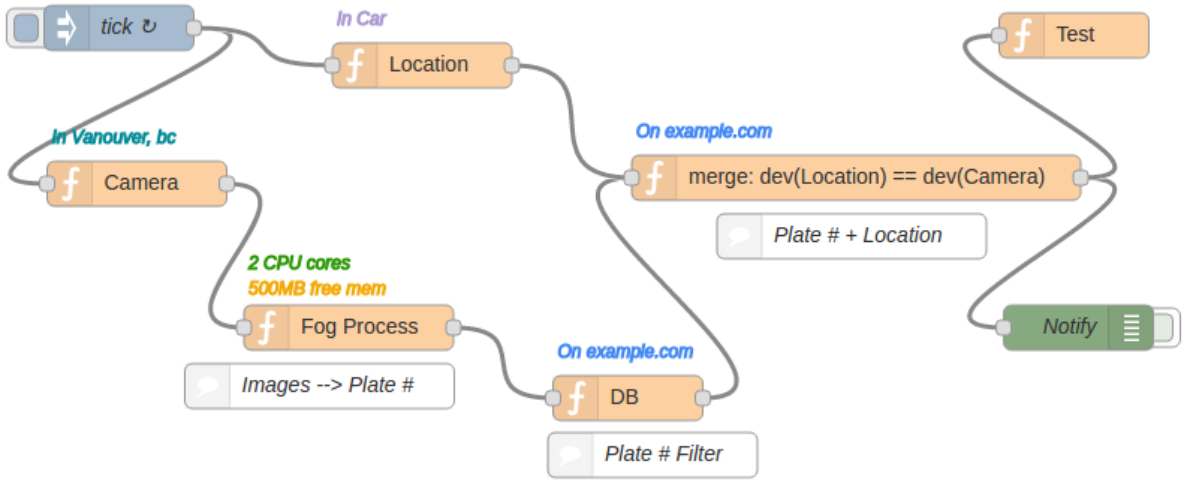


FIGURE 4: SPECIFIYING CONSTRAINTS ASSOCIATED WITH PROCESSING MODULES IN A DATAFLOW PROGRAM

The goal is to make the application model more suitable for a class of fog-based applications that are heavily dependent on the context associated with the edge devices they operate on. As a result, the developer can not only specify which type of device a node should run on (e.g mobile, server or laptop, etc.) but can further constrain where the node should run based on a variety of aspects such as memory size, processing capability, location etc. To address this need, we added support to allow application developers to specify these node constraints via the programming user interface. As can be seen in Figure 4 (above) a developer has associated a set of constraints, e.g. on server named **example.com**, or any device with **2CPU cores and 500MB of free memory.**

3.3.2 *Distribution architecture*

These capabilities allow us to support the edge processing needs of BigClouT. Once deployed, an application is constantly monitored and its constraints are evaluated. As the application's environment (or context) changes, the D-NR platform re-evaluates the constraints and redistributes processing across the fog network. This provides a dynamic edge processing capability that is not specifically programmed by BigClouT developers, but is derived implicitly from the constraints they define for an application.

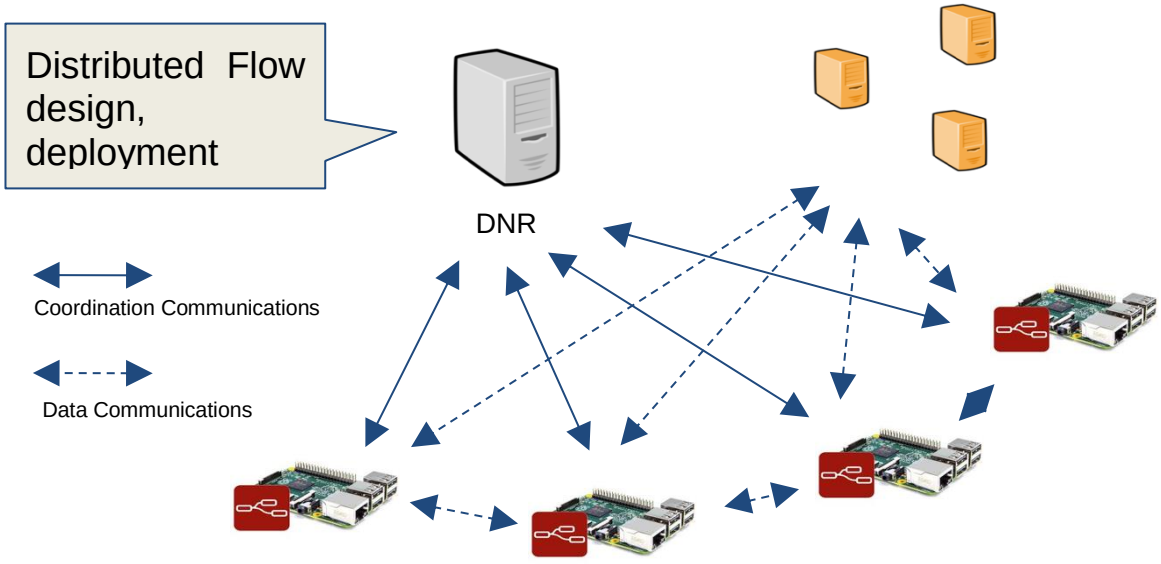


FIGURE 5: DISTRIBUTED COORDINATION VIA BROKERS MANAGING EDGE PROCESSORS

As it can be seen in Figure 5, this is accomplished by a set of coordinated brokers that work together to initially distribute applications and then to monitor them as they run and to make decisions on when to move edge processing.

4 CORE DEMONSTRATION

We demonstrate how BigClouT architecture and its implementation provides distributed city service which solves city management problem. This section describes our basic scenario firstly, then we describe its implementation and demonstration.

4.1.1 Road infrastructure monitoring – basic scenario

In Japan, the problem of aging road infrastructure is increasingly important since around half of the road network (including bridges) was originally built 40 years ago. To repair roads and road-related equipment such as mirrors and road markings, it is important to understand which area of the city has what kind/level of problem. Currently, the municipality just inspect road condition of limited areas (mainly national roads) once every several years. Thus, the condition of most roads in the city is not monitored.

During discussions with the road management section in Fujisawa city, we found that they need to specify priority for roads to be repaired. In addition, it is desirable to check actual road status visually to confirm the necessity of repair. To meet the requirements, we provided the following scenario which contains three phases of road infrastructure management. The scenario leverages garbage trucks as sensors, something that was demonstrated at the first review of the project. Figure 6 shows the overall scenario used in the demonstration which consists of three phases.

Phase1 Macro Sensing

Garbage trucks cover more than 98% of the road network in a week, so we can collect complete/full road health condition by attaching sensors and edge computational resources to garbage trucks. Uploading all of the road images is unrealistic because of limited network bandwidth. Thus, we analyse road status such as condition level with GPS coordination at edge-side (computer on garbage truck). The analysis leverages an edge analysis component called DeepOnEdge developed as part of WP3. Analysis results from all garbage trucks are collected and published to the BigClouT data warehouse by using the distributed data flow component called distributed Node-RED (D-NR) developed in WP2.

Phase2 Priority Decision

Once the road condition level of the whole city is determined, the priority for each road is determined, i.e. which roads must be repaired as soon as possible. For setting reasonable priority, in addition to road damage information, various conditions such as how many people/cars use the road should be considered. For combining and analysing different data, we utilise the big data analysis component called KNOWAGE developed in WP3. According to defined priority, micro sensing operation is sent to each garbage trucks by D-NR.

Phase3 Micro Sensing with Privacy Protection

To meet the requirement that city officers see the actual road image (i.e. a visual confirmation), the last phase is to upload actual image of roads to be repaired from garbage truck sensors. Image taken by garbage trucks may contain privacy information such as faces of pedestrians or car numbers. Therefore, such privacy information should be removed before sending the image.

Which means that "anonymisation" of the images must take place by using the DeepOnEdge component. Finally, anonymised images from specific roads are collected through D-NR, and city officers can make plans of road repairs.

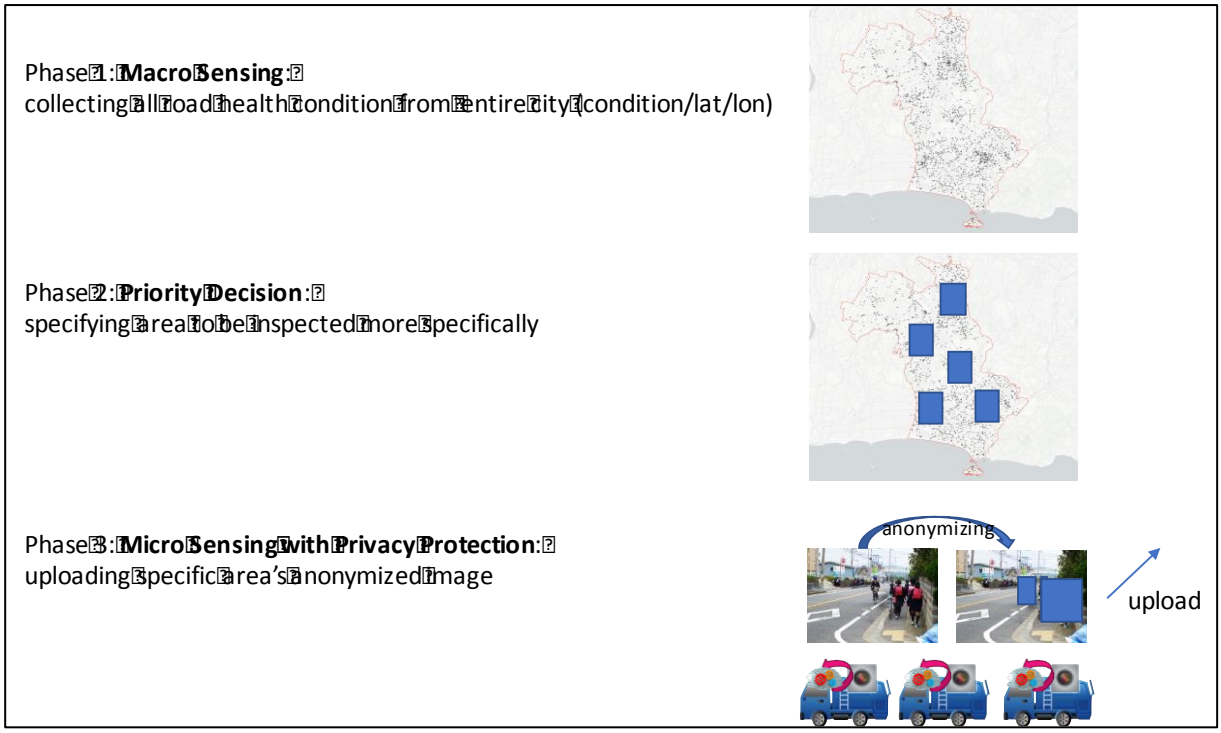


FIGURE 6: OVERALL FLOW OF ROAD MONITORING DEMONSTRATOR

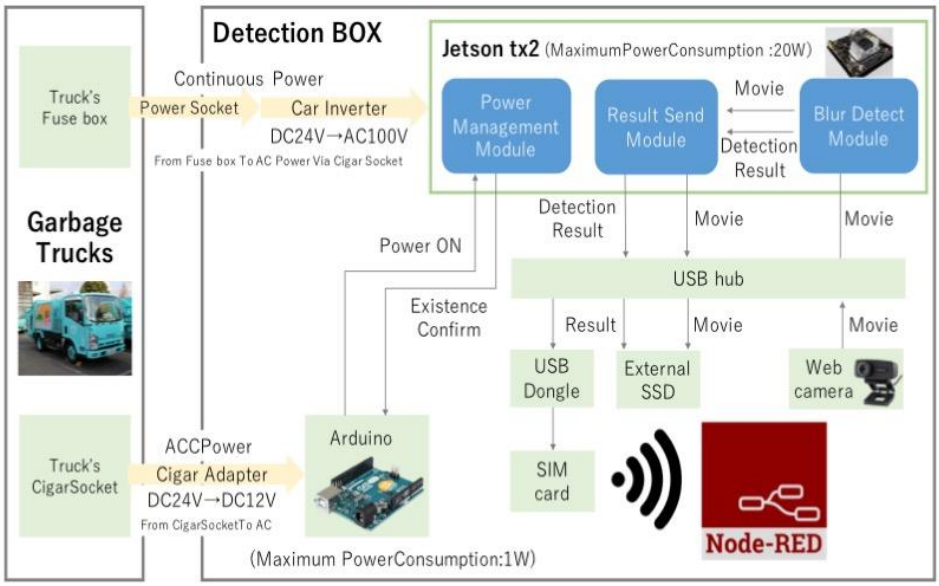
4.1.2 Road infrastructure sensing: Details

The scenario is detailed as follows. For the review, the demonstration will use a live network of devices, but will rely on captured movie/data replay to show live demonstration.

Phase1: Macro Sensing

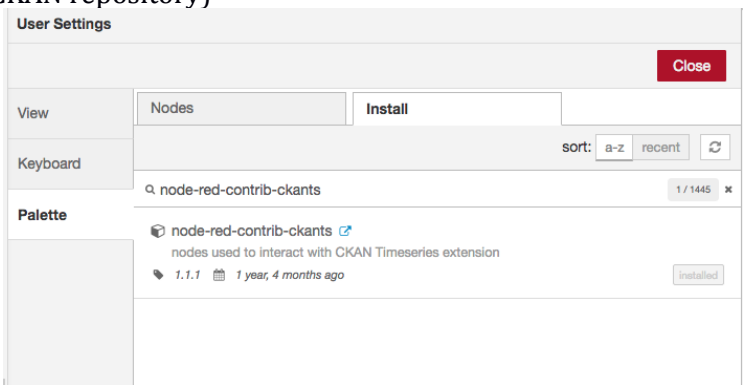
- Road damage detection on edge

➢ Introducing designed new edge hardware based on Jetson for deploying to actual garbage trucks



- Live preview of how DeepOnEdge detects several examples of road damage

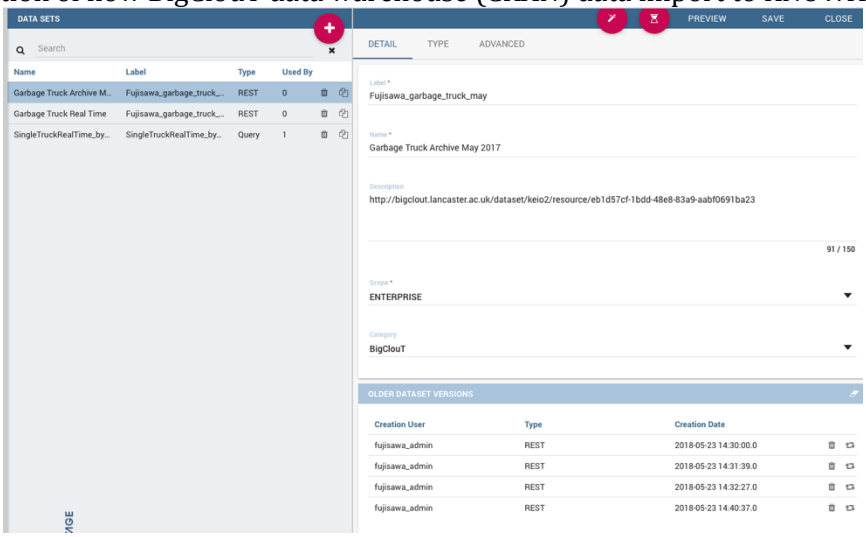
- Aggregating damage information from multiple trucks
- Presenting overall process flow defined by D-NR
- Demonstrating how the flow is deployed to each device
- Demonstrating how detected road information is aggregated and published to BigClouT data warehouse (CKAN repository)



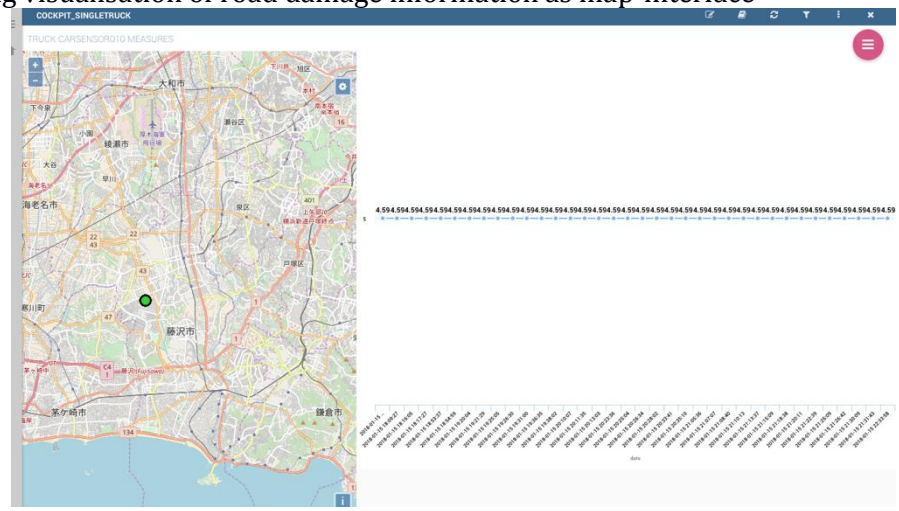
Phase2: Priority Decision

- Understanding road condition from whole city by KNOWAGE

➢ Demonstration of how BigClouT data warehouse (CKAN) data import to KNOWAGE



➢ Presenting visualisation of road damage information as map-interface



- Deciding priority by combining several city information

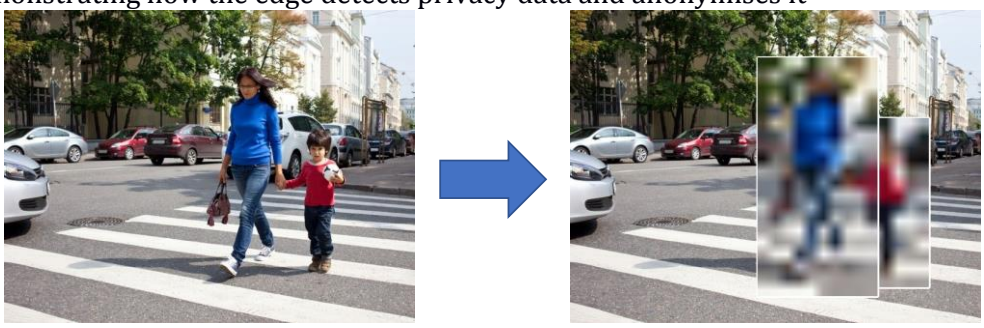
➢ Presenting how we analyse different city data to decide priority for repairing road

➢ Demonstrating how micro sensing operation is sent to each devices via D-NR

Phase3: Micro Sensing

- Image anonymisation on edge

➢ Demonstrating how the edge detects privacy data and anonymises it



- Anonymised image transfer

- Presenting how anonymised image can be transferred via D-NR

5 SUPPORT DEMONSTRATIONS

In addition to the core demonstration described above, we have two other demonstrators available which highlight other aspects of the visual programming tool D-NR.

5.1 Home sensing, energy management and pollution

In line with the Bristol city (BIO) scenarios, we will demonstrate a basic framework using the edge processing of BigClouT (based on D-NR) for distributed home sensing. This demonstration will include processing done at the sensor, within the premises and in the cloud. Driven by fine-grained information collected at a local level, this hierarchical (and distributed) processing architecture is able to make data-driven decisions at different layers. This includes real-time decisions made locally, such as alerting and analytics, as well as macro insights made across many different households. Potential sources of information may include home energy sensing, on a per-plug basis. This enables detailed usage information to be captured regularly. This can be used by the user to understand their consumption patterns, but also by local authorities and planners when designing for future growth and expansion in infrastructure.

Pollution sensing in the home is another potential aspect to be considered. By collecting information locally, householders can better understand the current levels of pollution within their own home and ensure that they are within reasonable tolerances. If limits are breached, then an alerting platform can be used to warn the user, who can then take appropriate measures to counteract any potential harmful side-effects. When this information is correlated across many households, local councils and governments can identify hotspots of pollution, and seek to understand the cause of these issues. Unlike a householder, they are empowered to regulate and curb such pollution causing practices and can therefore act to remedy the situation. In both cases, the distributed nature of the platform allows for a number of stakeholders to gain relevant and contextualised information in a meaningful and timely manner.

To realise this supporting demonstration, a small-scale deployment will be made. This includes the installation of both sensors and fog computing resources in a home environment, similar to the dwellings in which such a system would be likely deployed. Furthermore, rather than using a lab-based setup, this also provides genuine real-time data, allowing the end-to-end system to be tested in a meaningful and realistic manner.

Pollution monitoring is achieved using a Nova SDS011 Air Quality Sensor. This enables live monitoring of PM2.5 and PM10 values in the local atmosphere. These two metrics are key indicators of air quality and cover a wide-range of pollution types. When coupled with a high-level of accuracy and cost effectiveness, these sensors make an ideal candidate for mass deployment in homes and residencies. This sensor is directly connected to a Raspberry Pi, which periodically reads the various values of the sensor. Importantly, the Pi is acting as a D-NR node, and has Node-RED software running on it. This facilitates the direct reading of data into the platform.

Once this pollution data is ingested, it can be processed, manipulated and altered at any point. In this case, local processing will take place at the node. This can be configured from within the *Studio*

element of D-NR, and requires no specific configuration, nor explicit change, compared to building and deploying a stream in a non-distributed environment. The sensor-based (most local to the sensor itself) processing will be conducting basic averaging of the readings, and is used to provide a still frequent, but somewhat condensed, 10-second moving average of pollution levels in the immediate locality.

A similar deployment is used for the energy monitoring capability. Two through-plug energy monitors are deployed throughout the home. These report current usage statistics back to a Raspberry Pi using a proprietary radio technology. Data is again retrieved and ingested through the use of Node-RED. Once this data is ingested within the system, it is again parsed locally on the nearest connected D-NR node.

As with the pollution sensing, the node performs basic aggregation at this point, configured from a central point using the *Studio* element of D-NR. In both cases, only simple aggregation is conducted; this can be replaced with more sophisticated sampling and manipulation as required. This is where the constraint-based distribution features of D-NR are particularly appropriate: devices can be chosen based upon their capabilities, including processing power and memory. This ensures that intensive tasks are only completed on devices capable of doing so and avoids the unnecessary saturation of less-capable devices. Given that such limited devices these may be located close to the edge of a network (such a near to a sensor), D-NR allows for this capability to be considered when locating functionality.

Once both data types are ingested into D-NR, and pre-processed at source, the data can then be sent freely to other nodes, on which further actions can be achieved. For the purposes of this demonstration, the data will be sent immediately to a further D-NR node, operating still within the home environment. Whereas the other nodes are connected directly to sensors (of differing types), this node has no such connections, and is dedicated solely to processing the data collected within the home. As such, it will collate the information received from both sources, and provides a simple visualisation to the user. Furthermore, it stores the data for longer-term retention (resource permitting), allowing some historical analysis to take place if required.

It is this device that will conduct more sophisticated analysis of the data. Rather than sending the entire untouched dataset straight to a central repository (in this case, a D-NR node located in a cloud environment, to be discussed later), insights can be derived locally. Not only does this prevent bandwidth intensive traffic from traversing potentially limited Internet connections, it also maintains aspects of privacy, as this potentially sensitive data never leaves the users home. Furthermore, this approach reduces processing latency, and allows immediate actions to be taken if necessary.

More pertinently in the case of pollution, this may involve alerting householders when pollution levels are considered above normal, allowing citizens to take immediate action such as closing windows. In the case of this demonstration, this is realised by illuminating a simple LED light when these circumstances occur. This is again realised through the use of Node-RED, which allows interaction with the physical world too. In these instances, latency is critical; delay may cause inconvenience, or at worst lasting damage, if left unreported.

This demonstration focuses primarily on these two aspects: pollution and energy usage. However, the system can be easily and readily adapted to add new sensors. It is envisaged that this node,

located in the premises itself, will act as a gateway for the different sensing and control systems that may ordinarily, or at least in the future, be deployed in such a household.

The final location in which a node will be deployed is in a cloud-like environment. More specifically for the purposes of this demonstration, this involves hosting a D-NR node at Lancaster University. Although this demonstration is intentionally limited to a single-household for the purposes of supporting larger-scale endeavours, it is envisaged that multiple households would deploy a similar arrangement (in terms of pollution, energy and other such sensors).

In this configuration, each household (as the aggregating D-NR node within each) would then connect to a centralised point. Here, it would be possible to collate and aggregate data amongst these many different locations, which gives a much broader and wider view of the current state of each of these aspects. Given the richer capabilities located within a household (compared to current provision), further summarised data can be provided to this centralised node, at much longer timescales compared to those proposed for the home itself. Furthermore, this capability allows for anonymisation and privacy preserving processes to take place directly in the home; the centrally collected data is already in a state fit for sharing.

At this point, real-time alerting and visualisation is less important. Instead, longer-term trends, particularly those involving wider geographical locations, can be observed. Given the potential for considerable compute and storage capacity in this environment, where there are fewer limitations around device constraints, sophisticated and intensive analysis can be conducted. The ability to use large-scale storage arrays also enables the retention of data over much greater periods. This enables the investigation of historical trends, and more importantly, grants the ability to observe any improvements or degradation as the result of actions made at a much larger scale.

As with before, the household node will be directly wired to the cloud-based nodes. This can be achieved through wiring the functions together and allowing D-NR to determine the appropriate nodes on which to run each function, given the relevant constraints. This uses the same D-NR *Studio* interface, further demonstrating the capability of developing a coherent and combined data processing platform using a single familiar interface. It is also envisaged that this demonstration will highlight the advantages of locating and using resources in many different locations, but with the caveat that constraints must be effectively considered for this to be successful.

5.2 Traffic management in Smart Cities

In this demonstration we show the use of the BigClouT D-NR platform to support a large scale traffic management scenario in a large city. Basic traffic management is enabled by sensors around the city determining the levels of traffic flow through key intersections in the city. To support this, we use a D-NR application that is running at edge processors throughout the city and using cameras to monitor traffic. The camera monitoring uses image recognition techniques to count he cars passing through a junction and to determine average vehicle speed and make/model of vehicle.

The D-NR application makes use of the dynamic nature of the edge processing platform by using a basic self aware mechanism. As load on a particular sensor processing node increase, e.g. as traffic volumes increase and the processing required to track traffic this also increases, the application dynamically reconfigures itself to move processing load to other nodes in the network.

To demonstrate this dynamic edge processing capability, we visualise the internal processing of the D-NR platform to show the resources consumption of participating devices using the DNR-Studio tool so we can visualise the devices' load in real-time. For example, currently we have a device monitor tab in D-NR, we can show resource bars for each device, e.g. a CPU bar and a memory bar.

Secondly, we simulate a wide area city settings by running several participating Node-RED instances, in a laptop, and using the Jetson edge processing board which are installed in the Fujisawa garbage trucks.

The basic application flow consists of 3 basic nodes, camera capture, image extraction and vehicle recognition and identification. The third node is a complex AI algorithm that requires significant processing. Our demo deploys this flow, as a distributed application, to a server and 2 edge processors. Once the application is started, the D-NR edge processing platform monitors load on the overall network and automatically balances load by starting and stopping edge processors as load increases. For the demonstrator, we visualize the internal workings of the D-NR platform by showing load on the individual edge processors and overall identification results.

6 CONCLUSION

The distributed dataflow programming tool, D-NR has been developed as part of WP2, T2.4 and described in this document. To demonstrate its capability and integration into the BigClouT architecture we have described a core demonstrator and 2 support demonstrations that highlight features and capabilities of D-NR. These demonstrations illustrate a number of key points:

- Distributed data flow and managed edge processing
- Visual programming using flexible constraints to control edge processing
- Dynamic behaviour as the system responds to changes in the environment
- Integration with BigClouT data warehouse and WP3 components, Deep on Edge and KNOWAGE