Scalable Mobile Sensing for Smart Cities: The MUSANet Experience

Alexandre Meslin[1,2], Noemi Rodriguez[2], Markus Endler[2],
[1] Universidade Federal do Rio de Janeiro, Núcleo de Computação Eletrônica, Rio de Janeiro, RJ 21941-916 Brazil
[2] Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, Rio de Janeiro, RJ 22451-900 Brazil

In this paper, we present and analyze MUSANet, a hierarchical, distributed, context-aware architecture for collecting, processing, and distributing data in smart cities. We discuss some use case examples related to monitoring and predicting bus arrivals in public transportation and weather conditions. We also present performance results in different scenarios that point to the feasibility of our goal: a scalable architecture with a fast response time to traffic events. MUSANet is based on a three-tier architecture distributed over cloud, fog, and edge, and supporting complex event processing (CEP) in all of them. Although the system is under development using the InterSCity platform in the cloud, the ContextNet middleware at the fog, and the Mobile-Hub platform at the edge, the MUSANet architecture can be deployed using other platforms, maintaining the concept of tiering responsibilities to minimize network bandwidth and delay, group communication, and broad mobile support.

***Index Terms*—Smart City, Internet of Things, Fog Computing, Cloud Computing, Edge Computing, Middleware**

I. Introduction

Over the last years, we have witnessed the growth of the so-called Internet of Things: the original Internet has been extended to encompass arbitrary objects and things, and we now have a massive network of sensors and actuators. This new infrastructure allows users to interact with arbitrary objects over the Internet, receiving information, and also modifying the external state. Currently, the IoT refers to this vast range of items connected to the Internet. Connected objects have moved from inside homes and offices to the streets, creating the basis for smart cities. Today, cities have sensors capable of transmitting information to the public and the government in near real-time on a variety of items, such as the location of a particular bus, traffic conditions, air quality, availability of car parking spaces, as well as less visible services such as water or gas-leak monitors.

For a city to be considered a *smart city*, however, it is not enough for it to provide an extensive sensor network. Much more is needed, such as intelligence in the economy, governance, mobility, and environment, among others [1].

A smart city is made up not only of sensing devices and M2M transactions. Since the main actors of the city life are its inhabitants, and almost all of them carry a smartphone, data collected from these "human-centered" sensors may soon become the main contributors to the digital fabric that describes the city operation. Almost all inhabitants with a smartphone will generate data that must be processed or stored or both and will demand information on situations that are important to them according to their location, preferences, and context. This human interaction creates a new paradigm named Mobile Crowd Sensing (MCS) [2], [3].

Supporting mobility at this level entails managing connections to optimize response time and make efficient use of the communication network. When we consider people with their smartphones and other mobile devices and vehicles — buses, trucks, automobiles, and trains, — with the most varied types of sensors, we realize that software developed for modern smart cities must not only support a large number of connections but also support for roaming, allowing them to move through the city maintaining the best possible connection to the infrastructure.

One of the biggest challenges inherent in the project of a smart city is designing the flow of information, from the moment a sensor captures data to its final use. As discussed by Delicato and others [4], a typical IoT ecosystem is organized into three layers: (i) a bottom tier encompassing the devices and things, (ii) a top tier containing cloud nodes, and (optionally) (iii) a middle tier with smart gateways and edge nodes. The scalability, processing, and adaptation capacity of each of these layers will determine the effectiveness of a smart city ecosystem. For the case of smart cities, and specially to support mobile devices, it is essential to have a middle layer providing smart gateways that manage connections and process information. Mobile nodes depend on such gateways for their connection to the stationary Internet. An efficient infrastructure to support mobile users must migrate their connections among gateways to guarantee the connection quality between devices and gateways, and the load balance among the gateways. On the other hand, processing, filtering, and aggregating information as close as possible to the source is essential to eliminate unnecessary network traffic and reduce response times [5], [6].

Over the last years, several middlewares and platforms for smart cities have been proposed [7], [8], [9], [10], [11], [12], [13], [14]. However, they mostly focus on one of these tiers, in some cases providing entry slots for connection to other layers. Most of these platforms also do not have specific support for mobile sensing and actuation. In our work, we investigate how a three-tier architecture can provide support for scalability and flexibility in the provision of smart-city applications with support for crowdsensing and mobile users.

In previous work [15], we presented MUSANet, our three-tier architecture with support for mobile devices, and discussed scalability results relating to the number of supported connections and processing capability. Unlike most three-tier systems, MUSANet has brought CEP processing capabilities close to sensors and actuators, taking border computing to mobile devices, and thus reducing response time and network bandwidth consumption. MUSANet also uses a new concept in communication called *context-aware groupcast*, where messages can be sent to groups defined by custom rules.

In this paper, we study the behavior of MUSANet under several stress scenarios and show that the architecture behaves very well as regards throughput and scalability. We describe tests to determine the limits up to which the system can handle data. We also present and derive conclusions from a new set of tests in which we investigate the effect of data processing at the edge or fog. Most of the experiments use actual data from public transportation from the city of Rio de Janeiro, Brazil.

The remainder of this paper is organized as follows. Section II introduces MUSANet and describes the involved technologies. Section III presents the results of preliminary experiments. Section IV discusses related work. Finally, Section V contains some concluding remarks.

II. Overview of MUSANet and Enabling Technologies

This section provides an overview of the MUSANet[1] architecture, fully described in previous work [15], and the enabling technologies used in its implementation.

Our system architecture employs edge and fog computing to create a scalable infrastructure for smart cities with support for mobile nodes. MUSANet is a distributed multi-tier context-aware architecture for capturing, storing, and processing sensor data, sending data and commands to actuators, and receiving and publishing information through publish-subscribe protocols. We structured its architecture into three layers, as Figure 1 shows.

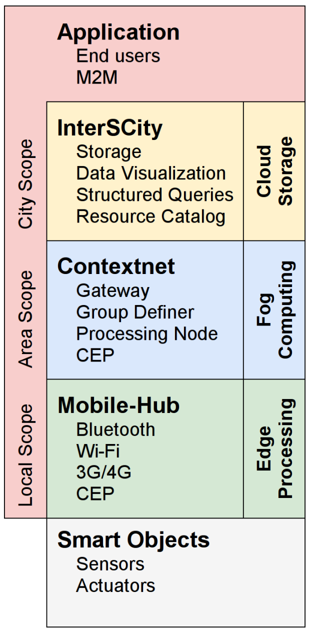

Figure 1. MUSANet 3-Layer Architecture.

[1]Available at https://github.com/meslin8752/InterSCity-onibus

The MUSANet top tier is based on a scalable storage ecosystem platform distributed in a cloud. In our implementation, we used InterSCity[2] [16], an open-source platform that provides basic blocks for the development of applications related to smart cities through REST APIs. This ecosystem stores data in a hierarchical way, allowing high-level queries. For instance, a bus may have some monitoring attributes, including its speed and geographical position, composed of latitude and longitude. One can send to this tier a *querystring* to locate all resources (buses) within a radius of 500 meters from a given point, as shown in Listing 1:

```
lat=-23.45&lon=-45.67&radius=500&
capability=bus_monitoring
```

Listing 1. A complex query to locate bus monitoring sensors.

The middle layer, implemented by ContextNet[3] [17] in the fog, includes interconnected gateway servers distributed throughout the city, allowing stationary or mobile devices to connect to the system. The proximity of a gateway to a mobile node decreases the response time to global or regional events, allowing the development of third-party systems with fast-response time for interactive or near real-time applications. A regional event would be the control of autonomous cars on the street, while global events would be monitoring flooding using data from the amount of rain, and river and tidal levels. ContextNet allows communication with edge devices via unicast (for a node individually) or groupcast (for a node group) messages. The ContextNet *GroupDefiner* module classifies edge nodes into groups using a custom rule. In the same network as the Gateways, we can deploy multiple Processing Nodes (PN), a GroupDefiner (GD) to manage mobile device groups, and a Points of Attachment Manager (PoA-Manager) to select the best gateway for each mobile device, all of them forming a slice of the ContextNet. A set of ContextNet slices forms the middle layer ContextNet fog.

At the bottom layer, the Mobile-Hub middleware [18], that runs on any Android device, including smartphones, can discover and connect to sensors and actuators, and function as a hub of data and commands to and from the middle layer. It also supports the online analysis and processing of the data streams received from any connected or embedded sensor. Embedded CEP[4] [19] (Complex Event Processing) analyzes events in near real-time, and the CEP output can trigger a local actuation or be sent to the middle layer with date and location context.

With the forthcoming implementation of 5G technology, mobile service providers will deploy virtual micro-clouds connected to their antennas. MUSANet is also well suited to such infrastructure, as it would be possible to deploy its middle-layer gateways in these micro-clouds. On the other hand, where public Wi-Fi is available, MUSANet gateways can be installed together with network access points. In both cases, the gateways will be deployed close to the sensor where they capture data.

[2]Available at https://gitlab.com/smart-city-software-platform/dev-env
[3]Available at http://www.lac.inf.puc-rio.br/dokuwiki/doku.php
[4]We use the engine from Espertech (http://www.espertech.com).

The three tiers of MUSANet allow edge nodes to process data and events with near-zero latency while the nearest ContextNet slice can process events relative to its region in near real-time. Information for historical or relevance analysis for all or most of the city can be stored in the cloud and processed by external applications.

III. Experiment Results

In this section, we describe experiments that examine the MUSANet throughput with a different number of gateways (subsection III-A); the time to send a message to an actuator depending on the distance they are from the source (subsection III-B); and the number of nodes (sensors and actuators) that can connect to a single gateway (subsection III-C). We conducted these experiments by running our infrastructure on a testbed composed of virtual machines hosted in the PUC-Rio virtual machine cloud. The environment consists of four virtual machines (VM001-VM004) set up as ContextNet slices. Each slice contains a Gateway, a Processing Node, a Group Definer, and a PoA-Manager, as well as a complete instance of InterSCity. We arranged these machines in separate networks connected by a virtual machine (VM005) with six network interfaces. We configured the four ContextNet network interfaces with delays compatible with what we would find on the actual Internet for more accurate experiments. InterSCity and ContextNet could share VM001 in the experiments we present in this paper because all results are related to the edge and fog performance. For InterSCity performance, please refer to Del Esposte [20]. Figure 2 shows the testbed architecture.

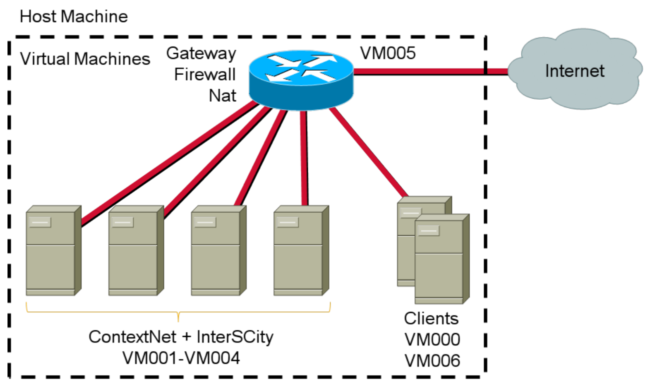

Figure 2. Testbed Architecture.

All information used to emulate the sensors is composed of real data obtained from the database of the city of Rio de Janeiro. We use data on buses from the city of Rio de Janeiro instead of traffic simulators to make the experiments more realistic. To emulate sensors or buses, we use two virtual machines (VM000 and VM006) connected to the same network. The use of a single network to emulate all buses does not interfere in the experiment because one client does not communicate directly with another client, only with the ContextNet slices. We created a separate thread to represent each bus that appears in the database. These threads connect to one of the ContextNet Gateways in the same way that an Android device running Mobile-Hub would do. All threads act independently, without sharing data, to emulate the actual system as closely as possible. Once connected, the thread begins to send data to stationary infrastructures, just as real buses would. For testing purposes, we control how often the threads emulating buses send data. All virtual machines have Internet access through VM005, which is in charge of isolating the actual Internet network traffic from the network traffic generated by our experiments.

A. Load and Scalability Test

We first performed an experiment to verify the maximum number of connected clients that each ContextNet Gateway supports, as well as to investigate possible scalability issues when increasing the number of Gateways. We performed two different types of tests. In the first set of tests, we emulated up to 8,000 buses in a single region, all connected to the same Gateway. In the second set, we connected up to 16,000 buses to up to four gateways in different networks to verify scalability.

Each emulated bus sends data containing its geographical position (latitude and longitude), its speed, its serial number, and its bus-line ID every 5 seconds, 1,000 times. The data received by the Gateway are then forwarded to the Processing Node of the slice to be processed or to be stored in InterSCity. These data are also checked by the GroupDefiner to map mobile nodes to the correct groups. We repeated this experiment 10 times and Figure 3 shows the average throughput achieved when connecting 1,000 to 8,000 emulated buses with a single ContextNet Gateway. For comparison, it also shows the theoretical maximum throughput.

We can determine the maximum number of packets per second mathematically by dividing the number of buses used in the experiment by the time in seconds between each packet transmission. The results obtained in the experiment were very close to those previously calculated with a maximum difference of 0.34%.

In this experiment, there was no significant loss of packets when using up to 10,000 connections. Because we extrapolated somewhat beyond the target number of connections per Gateway, we can notice that with up to 10,000 buses connected, there was no significant loss of packets, showing that a small amount of gateways is required considering only this amount of buses. When we try to emulate above 10,000 buses connected simultaneously to one gateway, connections start to become unstable, and many buses are disconnected. Moreover, when they try to reconnect, they cause an extensive sequence of connections/disconnections, preventing the experiment from continuing. For comparison purposes, we repeated the test using the same infrastructure with FogFlow [21] gateway. The results we obtained were very similar to those observed using ContextNet. We did not run the test with more than one FogFlow gateway because, to the best of our knowledge FogFlow has no load balancing or gateway switching mechanism for connected IoT devices as is the case with ContextNet PoA-Manager.

We next emulated the connection of up to 16,000 buses in up to 4 ContextNet slices. In this experiment, only one of the slices has a Processing Node, so the other slices had to send the data to the slice with the Processing Node.

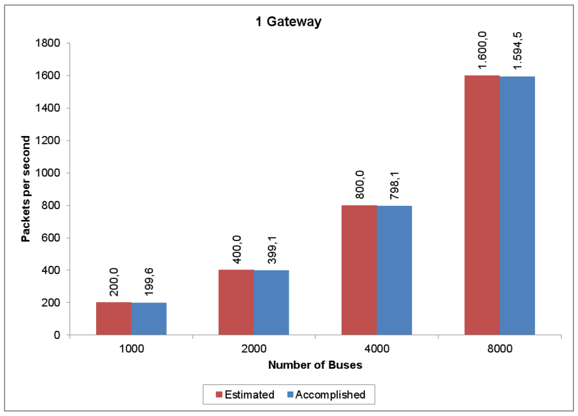

Figure 3. Load benchmark with 1 ContextNet Gateway for 1000, 2000, 4000 and 8000 buses.

For two slices, the observed results were again very close to the estimated values. Figure 4 presents these results, which demonstrate that even with this amount of clients, connections between the Mobile-Hubs running on emulated buses and the ContextNet slices occurred without problems.

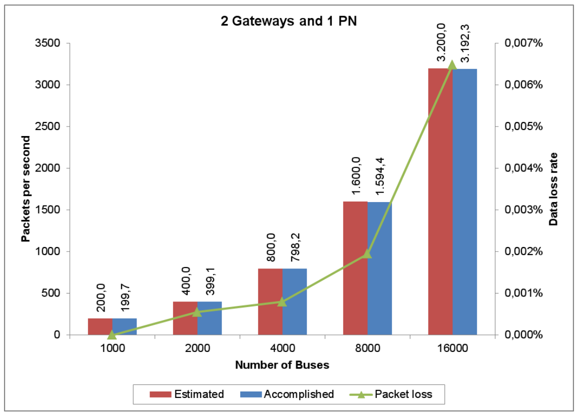

Figure 4. Load benchmark with 2 ContextNet Slices. The bars represent the throughput, and the line, the data loss rate.

We next examined data rates and data loss in the communication between buses and the Processing Node. Table I shows the results for 1 to 10 ContextNet slices. In each case, only one of the slices contained a Processing Node and an InterSCity installation. In experiments with more than one slice, we distributed the buses among ContextNet Gateways. Slices that did not have a Processing Node sent the received data to the slice with a Processing Node. Up to 16,000 buses were emulated, with each bus sending data every 5 seconds. Total losses include data lost during transmission between the bus and Gateway and between the Gateway of one slice and the Processing Node of another slice. Despite the increase in data losses with an increasing number of slices, we can observe that the percentage of data lost concerning the amount of transmitted data is always low, not exceeding 0.007%, considering the number of connected devices equivalent to the bus fleet for each gateway.

Referring to the city of Barcelona [22] in Spain, where 1,800 sensors generate 1,300,000 pieces of data per day [23], i.e., 1.5 pieces of data per second and scaling to a city the size of Rio de Janeiro, which has an area 16 times larger, we can assume that 28800 sensors would generate approximately 24 pieces of data per second. In our experiments, we generate data from 8,000 buses every 5 seconds, i.e., 1,600 pieces of data per second. These experiments show that even for a large city such as Rio de Janeiro, São Paulo, or Mexico, MUSANet supports the workload without significant loss of performance or packages. The number of gateways used in the deployment of the system is more dependent on the need to create regions or slices of ContextNet to always provide a point of attachment near the sensors than on the number of buses or sensors in the city.

Table I
DATA RATE FOR 1000, 2000, 4000 AND 8000 BUSES CONNECTED TO THE CONTEXTNET. THE MAXIMUM THROUGHPUT FOR 1000, 2000, 4000, AND 8000 BUSES ARE 200, 400, 800, AND 1600 PACKETS PER SECOND RESPECTIVELY.

Buses	1 GW	2 GW	4 GW	6GW	8GW	10 GW
1000	199.6	199.5	199.4	199.6	199.3	199.3
2000	399.0	398.7	398.8	398.9	398.6	398.4
4000	797.8	796.6	797.5	797.7	797.0	797.1
8000	1593.2	1594.4	1594.4	1594.8	1591.4	1593.2

B. *Delay from trigger to Notification*

In this experiment, we try to identify the time elapsed between the occurrence of an event and the notification of mobile nodes within the area of interest for this event. We compare the time it takes to notify mobile nodes connected to the ContextNet slice where the event occurred to the time it takes to notify mobile nodes connected to other slices but with interest in the same event. We use this experiment to demonstrate the importance of selecting the most suitable Gateway for each mobile node.

To make the experiment scenario realistic, we added delays of the order of 100 ms in the network interfaces of the virtual machine that has routing function (VM005) as Figure 5 shows. These delays are compatible with those that may be found on the Internet, as measured, for instance, in the path between the sensors and the City Hall website in Rio de Janeiro.

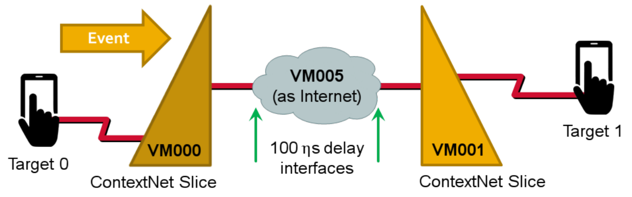

Figure 5. Event inserted through slice VM000 and user in different slices.

For this experiment, we created groups in GroupDefiner representing areas of 100 m^2 around the bus stops, similar to those illustrated in Figure 6. We also created groups representing areas along the way that the bus must cross to get to the bus stop. The distance between these areas should be calculated based on the time the bus takes to leave one area and reach another. This time must be sufficient to warn passengers at the bus stop. Listing 2 displays the CEP rule that we used in this experiment. The variable `regionNumber` represents the number of the area near the bus stop that will trigger the event `EventRegion` when the bus enters it.

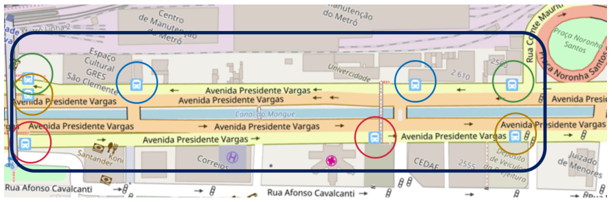

Figure 6. Bus stops and their microregions within a broader region.

```
    select * from EventRegion having
        EventRegion.region = regionNumber
```

Listing 2. EsperTech CEP code to detect that a bus is arriving at a bus stop.

We repeat this experiment 10 times and Table II presents the average delay for notifications to reach two targets, with Target 0 connected to the gateway selected by the PoA-Manager according to Target 0 geographic position, that is, even if it has moved through the city, it is currently connected to the gateway whose network latency to ContextNet is as small as possible. The second target, Target 1, is connected at any gateway other than the optimal gateway. Each target in this experiment represents a passenger at a bus stop waiting for their bus. In this case, the difference between the average notification time for Target 1 and Target 2 was 118 ms, about 31.30% slower. For applications involving people such as the one involving bus arrivals discussed here, we can ignore this difference since a person would not be able to notice it, but if we consider applications involving only machines, that time can be considerable. For example, let us consider an application where autonomous vehicles are traveling on a road. An application can coordinate the movement of vehicles on the road while connected to MUSANet, informing them of the current state of traffic lights and when their status will change. Also, each vehicle can inform the infrastructure of its intentions, such as turning right or left at the next crossing street. An essential and critical warning could be the need for sudden braking due to an unexpected obstacle like a pedestrian or animal crossing the road. The coordinating application can inform the other vehicles that are behind to brake as well upon being notified of the need for breaking the vehicle. If we consider vehicles at a speed of 60 km/h, a delay of 118 ms allows the vehicle to travel another 2 meters approximately, which could cause a collision. In many situations, vehicle-related decisions [24] need to be made in near real-time. Other applications in areas such as augmented reality, interactive games, and smart grid [25], also need a low latency between their servers and the final users.

C. Maximum Connections per Gateway

We next conducted tests to determine the relationship between the number of supported connections and memory usage. The graph in Figure 7 shows the number of nodes that could connect to a single gateway by varying the amount of available memory. On the X-axis, we have the amount of available memory in bytes, ranging from 2MB to 8GB. The Y-axis displays the number of nodes that successfully connected to the gateway. Through this graph, we can see that the number of possible connections in a single Gateway remains proportional to the amount of available memory up to approximately 8,000 connections. From this point, the curve tends asymptotically to 10,000 connections.

Table II
DELAY BETWEEN AN EVENT AND THE MOBILE NODE NOTIFICATION.

Target	Delay
Target 0	377 ms
Target 1	495 ms

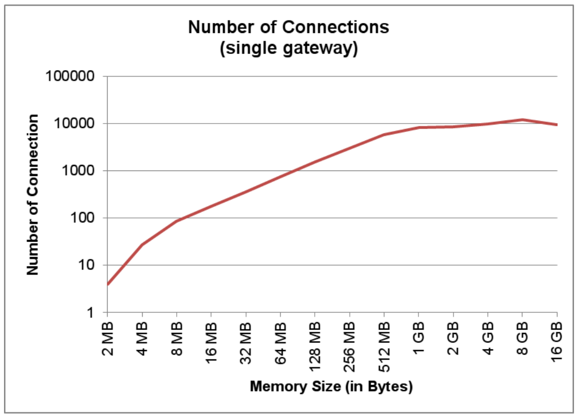

Figure 7. The number of connections accepted by a single ContextNet Gateway with a different amount of available memory.

We believe that this behavior is due to TCP limitations because the protocol provides less than 64k ephemeral ports for all applications running in the same machine. Once allocated, the port remains unavailable for a while, even when released.

IV. Related Work

We divide this section into two subsections. First, in Subsection IV-A, we present some applications that we could use to substitute some parts or layers of MUSANet or even could enhance some of its features. Then, in Subsection IV-B, we compare some smart city platforms with MUSANet.

All analyzed solutions have some desired characteristics [26] as scalability and security in common. Endler et al. [27] present more details about the implementation of security in MUSANet. Besides these characteristics, a framework for smart cities must also provide support for mobile nodes, allowing them to connect to a convenient gateway, and seamless disconnection and reconnection during roaming. Another essential point is decentralizing the cloud, distributing the processing capability in the fog, edge, and cloud, helping the system scalability [28] and also contributing to faster response to events. For example, a system that deploys some of its servers near the point of user access, like 4G/5G antennas or Wi-Fi access points, can provide a better user experience for augmented reality or interactive games. Even when we deploy the system in the cloud, if it can count with a distributed set of gateways, it will be less prone to network congestion. Other desirable characteristics for a smart-city framework may be group communication facilities, the type of its license, and the capability to process complex events.

A. Alternative MUSANet components

The MUSANet architecture is not dependent on the technologies selected in Section II, and we can use other technologies at each level of our system.

For example, the Dimmer [7] platform, which is a microservice-based platform that enables device and capability discovery, and provides an API for access to near real-time and historical data, can be used to manage the database. Its Service Platform can process data from sensors at the cloud like the MUSANet Processing Node does in the fog. The Dimmer gateways can gather data from a heterogeneous set of sensors, but its gateways are not able to preprocess data neither be installed at the network edge. Therefore, if we use Dimmer as the MUSANet upper layer, ContextNet and Mobile-Hub as middle and lower layer, we will add processing capability to the upper layer, and this new approach, compared to a solution using only Dimmer, will be able to process data at the edge, fog, and cloud, will have distributed low-latency gateways, and will support mobile sensors and actuator due to MUSANet PoA-Manager.

Another example is the FIWARE [8] platform, an OpenStack and Docker-based environment that stores data hierarchically in a MongoDB database, supporting both sensors and actuators. The platform also provides support for authenticating users and setting user roles with configurable permissions. Like InterSCity[5], the platform natively supports complex event analysis through CEP for near real-time event handling. Unlike MUSANet, FIWARE does not provide direct support for load balancing on its gateways, and the smart objects must insert all context information such as location, date of acquisition/generation, and so on, before sending data to FIWARE. However FIWARE was designed to be integrated into a myriad of platforms “Powered by FIWARE”, so, if we use it as the MUSANet upper layer, we will obtain a scalable and elastic system able to process data streams at the cloud thanks to the FIWARE CEP engine. MUSANet will provide, for FIWARE, gateways and processing capability in the fog for near real-time responses and sensors discovery, data acquisition and stream processing at the edge, very close to the data source (the sensors) due to Mobile-Hub features.

The Smart Streets IoT Hub described by Blackstock et al. [29] and the gateway presented by Aloi et al. [30] can be considered two alternatives for using Mobile-Hub. The first one was developed to interface with the Hyper/Cat platform [31]. The latter is a smartphone gateway with support for Wi-Fi, Bluetooth, LTE (H+, 3G/4G/5G, or else) networks and even ZigBee via a Micro-SD card from Spectec[6]. All of their features are implemented by Mobile-Hub, which is compatible with PoA-Manager to select the best gateway, even while roaming.

Amazon[7] offers a set of ready-to-use solutions for IoT. Programmers can customize their application using AWS Lambda, Amazon EC2, and others, for their individual needs. AWS Lambda, a serverless event-driven platform at the cloud, can process complex events. Although we can easily integrate different solutions to create a full IoT custom application scalable and elastic, there may be a network delay at about 130 ms between the data source and the platform at the Cloud because not all locations have an AWS datacenter nearby (in terms of network hops). Even if we implement a solution using the ContextNet and the Mobile-Hub at the fog and the edge, respectively, these delays will still be present.

B. Some Smart Cities Platforms

There are several surveys on smart city and IoT platforms [32], [33]. In this section, we focus on multi-tiered approaches.

The smart cities of Santander [34], Fujisawa, Mitaka, and Genova use ClouT [10], a hierarchical three-tiered platform, in their implementations Its lower layer can receive sensor data and send commands to WSAN actuators. The top layer allows access to the collected data. A third layer addresses security and dependability issues on the platform. ClouT uses the Esper CEP engine to process complex events in the cloud. More details about CEP at ClouT can be obtained in the ClouT Report [35]. Although the lower layer has processing capacity, data is sent to the processing and storage unit without any prior processing, while MUSANet can process events at any level.

IBM Watson IoT Platform [11], Microsoft Azure [12], Amazon AWS, and Google Cloud IoT Platform provide a cloud-based IoT platform capable of meeting smart city data processing and storage needs. Several components like storage and database, analytics and artificial intelligence, Complex EVent Processors, and IoT hub can be linked together for near real-time analysis of data streams, forming a middleware for smart cities.

IBM and Microsoft solutions require a third-party IoT gateway for sensors that cannot interact directly with the Internet and also do not provide edge data processing, i.e., computing capability close to sensor data acquisition. AWS Greengrass and Google IoT Android Things add processing capability to the edge, but none of them look for the best gateway to connect to the cloud. Their IoT components have multiple modules that support several sensors, but the vast majority of modules require machines with many computational resources. The support for the HTTPS, AMQP, AMQP over WebSockets, MQTT, and MQTT over WebSockets protocols allows us to connect a range of IoT devices (sensors and actuators) over the Internet. To the best of our knowledge, none of them provides roaming facilities for mobile devices. It is also important to note that the use of MUSANet ContextNet at the middle layer enables group communication, in addition to fog processing capability, besides network low latency gateways for mobile devices at the edge.

[5]The current InterSCity version now supports CEP.

[6]http://www.spectec.com.tw/sdz-530.html

[7]https://aws.amazon.com/

The Stack4Things [14] framework is a scalable system based on OpenStack [36]. The system has two layers. The bottom layer (IO layer), composed of sensors connected to Arduino devices [37], collects data and sends them to the upper layer, which is distributed in the cloud and provides an infrastructure as a service (IaaS) interface to the application level. The application level is not part of the Stack4Things framework. At the Stack4Things upper layer, there is support for Complex Event Processing (CEP) and application-level code injection. The lower layer uses Arduino YUN equipped with Ethernet or Wi-Fi interfaces to allow sensor devices to send, via the Internet, their generated data to the upper layer. The use of Arduino to interface the sensors allows programmers to implement the control logic of the sensors in C/C ++ language but limits the acquisition of data only to sensors with Arduino. As far as we know, the system does not provide support for mobile sensors and actuators.

In our architecture, we distribute processing into three levels as follows: (1) at the point of collection, close to the sensor; (2) at the interface between the sensors and the system core; and (3) in the system core. The lowest level, where Android devices collect sensor data and send commands to the actuators, can make some data processing at the edge of the network. This capability allows preprocessed, filtered, and aggregated information to be quickly delivered to and processed by MUSANet, differently from how it is implemented in Barcelona and Santander, where the gateways work as simple bridges between the WSN protocols and TCP/IP. Our work has an intersection with Participatory Sensing [38] by using smartphones or smart mobile devices to collect and analyze sensor data before sending them to the core of the infrastructure in the cloud. This preprocessing allows us to decrease the amount of data that is transferred over the network while enabling faster responses to local events. Unlike the architectures presented in this section, in MUSANet, using the Mobile-Hub and the data processing capacity of its host mobile device, the sensors are not just passive agents or simple data generators but units able to perform some local preprocessing of collected data. Consolidated information from mobile devices is transmitted with context data through the cellular network infrastructure to the various ContextNet attachment points (Gateways). While the lower level implements computing at the edge of the network, acting as a dual-stack gateway converting WSN protocols to TCP/IP, the top two levels are implemented in a fog/cloud, taking advantage of the existing Internet infrastructure.

V. Final Considerations

In this work, we presented a distributed framework for smart cities, able to process data close to sensors and distributed in the fog while data are stored in the cloud. Simulations show that the system is scalable, and the combination of multiple processing nodes and gateways to connect mobile devices at the edge to the stationary infrastructure in the fog significantly decreases response time. In a deployment in a densely populated environment of sensors and users such as hospitals, stadiums, and airports, the PoA-Manager can distribute smart objects connections to the nearest gateways, mitigating congestion or disconnections when accessing ContextNet.

A new ISO 37120 standard for Resilient Cities[8] defines a set of indicators and methodologies that can be used as a reference for establishing the data that should be provided as a basis for these and other applications. We intend to explore these application areas to evaluate MUSANet not only as regards scalability but also as regards the ease of developing different, third-party applications using the same set of data and interfaces.

Acknowledgement

This research is part of the INCT of the Future Internet for Smart Cities funded by CNPq, proc. 465446/2014-0, CAPES, proc. 88887.136422/2017-00, and FAPESP, proc. 2014/50937-1 and 2015/24485-9.

References

[1] R. Giffinger, C. Fertner, H. Kramar, R. Kalasek, N. Pichler-Milanovic, and E. Meijers, "Smart Cities - Ranking of European medium-sized cities," Vienna University of Technology, Vienna, Technical Report, 2007.

[2] R. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39, Nov. 2011. [Online]. Available: http://ieeexplore.ieee.org/document/6069707/

[3] A. Corradi, G. Curatola, L. Foschini, R. Ianniello, and C. R. De Rolt, "Smartphones as smart cities sensors: MCS scheduling in the ParticipAct project," in *2015 IEEE Symposium on Computers and Communication (ISCC)*. Larnaca: IEEE, Jul. 2015, pp. 222–228. [Online]. Available: http://ieeexplore.ieee.org/document/7405520/

[4] F. C. Delicato, P. F. Pires, and T. Batista, *Resource Management for Internet of Things*. Springer, 2017.

[5] L. Gupta, R. Jain, and H. A. Chan, "Mobile Edge Computing - an important ingredient of 5g Networks," Mar. 2016. [Online]. Available: http://sdn.ieee.org/newsletter/march-2016/mobile-edge-computing-an-important-ingredient-of-5g-networks

[6] A. H. H. Ngu, M. Gutierrez, V. Metsis, S. Nepal, and M. Z. Sheng, "IoT Middleware: A Survey on Issues and Enabling technologies," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 1–1, 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7582463/

[7] A. Krylovskiy, M. Jahn, and E. Patti, "Designing a Smart City Internet of Things Platform with Microservice Architecture," in *2015 3rd International Conference on Future Internet of Things and Cloud*. Rome, Italy: IEEE, Aug. 2015, pp. 25–30. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7300793

[8] FIWARE Foundation e.V., "FIWARE | Open Source Platform for the Smart Digital Future," 2018. [Online]. Available: https://www.fiware.org/

[9] J. Varia and S. Mathew, "Overview of Amazon Web Services," Amazon, Tech. Rep., 2014.

[10] J. A. Galache, T. Yonezawa, L. Gurgen, D. Pavia, M. Grella, and H. Maeomichi, "ClouT: Leveraging Cloud Computing Techniques for Improving Management of Massive IoT Data," in *2014 IEEE 7th International Conference on Service-Oriented Computing and Applications*. Matsue, Japan: IEEE, Nov. 2014, pp. 324–327. [Online]. Available: http://ieeexplore.ieee.org/document/6978633/

[11] IBM, "IBM Cloud," 2014. [Online]. Available: https://cloud.ibm.com

[12] Microsoft, "Azure IoT Edge documentation," Apr. 2019. [Online]. Available: https://docs.microsoft.com/en-us/azure/iot-edge/

[13] Google, "Cloud Computing Services | Google Cloud," Apr. 2008. [Online]. Available: https://cloud.google.com/

[14] D. Bruneo, S. Distefano, F. Longo, G. Merlino, A. Puliafito, V. D'Amico, M. Sapienza, and G. Torrisi, "Stack4things as a fog computing platform for Smart City applications," in *2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. San Francisco, CA, USA: IEEE, Apr. 2016, pp. 848–853. [Online]. Available: http://ieeexplore.ieee.org/document/7562195/

[8]URL:https://www.iso.org/files/live/sites/isoorg/files/archive/pdf/en/37120_briefing_note.pdf

[15] A. Meslin, N. Rodriguez, and M. Endler, "A Scalable Multilayer Middleware for Distributed Monitoring and Complex Event Processing for Smart Cities," in *2018 IEEE International Smart Cities Conference (ISC2)*. Kansas City, MO, USA: IEEE, Sep. 2018, pp. 1–8. [Online]. Available: https://ieeexplore.ieee.org/document/8656961/

[16] D. M. Batista, A. Goldman, R. Hirata, F. Kon, F. M. Costa, and M. Endler, "InterSCity: Addressing Future Internet Research Challenges for Smart Cities," in *Network of the Future (NOF), 2016 7th International Conference on the*. Buzios, Brazil: IEEE, 2016, pp. 1–6. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/7810114/

[17] M. Endler, G. Baptista, L. Silva, R. Vasconcelos, M. Malcher, V. Pantoja, V. Pinheiro, and J. Viterbo, "ContextNet: context reasoning and sharing middleware for large-scale pervasive collaboration and social networking," in *Proceedings of the Workshop on Posters and Demos Track*. Lisbon, Portugal: ACM, 2011, p. 2.

[18] L. E. Talavera, M. Endler, I. Vasconcelos, R. Vasconcelos, M. Cunha, and F. J. d. S. e Silva, "The mobile hub concept: Enabling applications for the internet of mobile things," in *Pervasive computing and communication workshops (PerCom workshops), 2015 IEEE international conference on*. St. Louis, MO, USA: IEEE, 2015, pp. 123–128. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7134005

[19] D. Luckham, *The power of events: An introduction to complex event processing in distributed enterprise systems*. Boston: Springer, 2008.

[20] A. M. Del Esposte, F. Kon, F. M. Costa, and N. Lago, "InterSCity: A Scalable Microservice-based Open Source Platform for Smart Cities:," in *Proceedings of the 6th International Conference on Smart Cities and Green ICT Systems*. Porto, Portugal: SCITEPRESS - Science and Technology Publications, 2017, pp. 35–46. [Online]. Available: http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0006306200350046

[21] B. Cheng, G. Solmaz, F. Cirillo, E. Kovacs, K. Terasawa, and A. Kitazawa, "FogFlow: Easy Programming of IoT Services Over Cloud and Edges for Smart Cities," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 696–707, Apr. 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8022859/

[22] T. Bakıcı, E. Almirall, and J. Wareham, "A smart city initiative: the case of Barcelona," *Journal of the Knowledge Economy*, vol. 4, no. 2, pp. 135–148, 2013.

[23] A. Sinaeepourfard, J. Garcia, X. Masip-Bruin, E. Marín-Tordera, J. Cirera, G. Grau, and F. Casaus, "Estimating Smart City sensors data generation," in *Ad Hoc Networking Workshop (Med-Hoc-Net), 2016 Mediterranean*. Vilanova i la Geltru, Spain: IEEE, 2016, pp. 1–8. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7528424

[24] The Economist, "Michael Dell plots his return to the public market - The opening Dell," Dec. 2018. [Online]. Available: https://www.economist.com/business/2018/12/08/michael-dell-plots-his-return-to-the-public-market

[25] R. Morabito, V. Cozzolino, A. Y. Ding, N. Beijar, and J. Ott, "Consolidate IoT Edge Computing with Lightweight Virtualization," *IEEE Network*, vol. 32, no. 1, pp. 102–111, Jan. 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8270640/

[26] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke, "Middleware for Internet of Things: A Survey," *IEEE Internet of Things Journal*, vol. 3, no. 1, pp. 70–95, Feb. 2016. [Online]. Available: http://ieeexplore.ieee.org/document/7322178/

[27] M. Endler, A. Silva, and R. A. Cruz, "An Approach for Secure Edge Computing in the Internet of Things," in *2017 1st Cyber Security in Networking Conference (CSNet)*. Rio de Janeiro, Brazil: IEEE, Oct. 2017. [Online]. Available: 10.1109/CSNET.2017.8241993

[28] J. Dizdarević, F. Carpio, A. Jukan, and X. Masip-Bruin, "A Survey of Communication Protocols for Internet of Things and Related Challenges of Fog and Cloud Computing Integration," *ACM Computing Surveys*, vol. 51, no. 6, pp. 1–29, Jan. 2019. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3303862.3292674

[29] M. Blackstock and R. Lea, "IoT interoperability: A hub-based approach," in *2014 International Conference on the Internet of Things (IOT)*. Cambridge, MA, USA: IEEE, Oct. 2014, pp. 79–84. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7030119

[30] G. Aloi, G. Caliciuri, G. Fortino, R. Gravina, P. Pace, W. Russo, and C. Savaglio, "Enabling IoT interoperability through opportunistic smartphone-based mobile gateways," *Journal of Network and Computer Applications*, vol. 81, pp. 74–84, Mar. 2017. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1084804516302405

[31] R. Lea, "Hypercat: an iot interoperability specification," Lancaster University, Book/Report/Proceedings 69124, Apr. 2014.

[32] A. Salami and A. Yari, "A framework for comparing quantitative and qualitative criteria of IoT platforms," in *2018 4th International Conference on Web Research (ICWR)*. Tehran: IEEE, Apr. 2018, pp. 34–39. [Online]. Available: https://ieeexplore.ieee.org/document/8387234/

[33] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015. [Online]. Available: https://ieeexplore.ieee.org/document/7123563/

[34] L. Sanchez, L. Muñoz, J. A. Galache, P. Sotres, J. R. Santana, V. Gutierrez, R. Ramdhany, A. Gluhak, S. Krco, E. Theodoridis, and D. Pfisterer, "SmartSantander: IoT experimentation over a smart city testbed," *Computer Networks*, vol. 61, pp. 217–238, Mar. 2014. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1389128613004337

[35] C. Greco, P. Paviac, M. Maggio, L. Gürgen, C. Munilla, T. Yonezawa, K. Tei, F. Ishikawa, and J. A. Galache, "D3.3 – CPaaS specification and reference implementation – second release," CEA, ENG, UC, ST, SAN, GEN, NTTE, NTTRD, KEIO, PANA, NII, EU and Japan, Tech. Rep. D3.3, 2013. [Online]. Available: http://clout-project.eu/wp-content/uploads/2013/05/ClouT_Deliverable_3.3.pdf

[36] OpenStack, "Build the future of Open Infrastructure," Oct. 2010. [Online]. Available: https://www.openstack.org/

[37] Arduino, "Arduino - Home," 2005. [Online]. Available: https://www.arduino.cc/

[38] F. Restuccia, S. K. Das, and J. Payton, "Incentive Mechanisms for Participatory Sensing: Survey and Research Challenges," *CoRR*, vol. abs/1502.07687, pp. 1–38, 2015. [Online]. Available: http://arxiv.org/abs/1502.07687

Alexandre Meslin Alexandre Meslin holds a degree in Electronic Engineering from UFRJ, and an MSc in Systems and Computer Engineering from the same University. He is currently working for a PhD degree in Computer Science at PUC-Rio. He is Systems Analyst at the UFRJ, lecturer at the PUC-Rio and Instructor at CEDERJ. Alexandre's main interests and experience are in Computer Systems, with focus on IoT, WSN, and smart city.

Noemi Rodriguez Associate professor at the Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), where she also completed her masters and doctorate degrees in Informatics. Her interests focus on distributed and concurrent programming, with an emphasis on the role programming languages have to play in this area.

Markus Endler Dr. Markus Endler is an associate professor at PUC-Rio, and founder and head of the Laboratory for Advanced Collaboration (LAC). He obtained his doctoral degree from GMD Institute and TU Berlin (1992), and the Livre Docente title (Habilitation) from the University of São Paulo (2001). So far, he has supervised 13 doctoral thesis, 29 M.Sc. dissertations, and co-authored more than 190 articles published in international journals and refereed conferences, as well as seven book chapters. His main research interests include distributed, mobile and pervasive computing, context-awareness, mobile coordination, Internet of Things, and Smart Cities.