Research online Publishing
www.ronpub.com

Open Access

Open Journal of Internet of Things (OJIOT)
Volume 4, Issue 1, 2018
http://www.ronpub.com/ojiot
ISSN 2364-7108

Past, Present and Future of the ContextNet IoMT Middleware

Markus Endler[A], Francisco Silva e Silva[B]

[A] Departamento de Informática, PUC-Rio, Rua Marques de São Vicente 225, Rio de Janeiro, Brazil, endler@inf.puc-rio.br
[B] Laboratório de Sistemas Distribuídos Inteligentes, PPGCC-PPGEE/UFMA São Luiz, Brazil, fssilva@lsdi.ufma.br

Abstract

The Internet of Things with support to mobility is already transforming many application domains, such as smart cities and homes, environmental monitoring, health care, manufacturing, logistics, public security etc. in that it allows to collect and analyze data from the environment, people and machines, and to implement some form of control or steering on these elements of the physical world. But in order to speed the development of applications for the Internet of Mobile Things (IoMT), some middleware is required. This paper summarizes seven years of research and development on the ContextNet middleware aimed at IoMT, discusses what we achieved and what we have learned so far. We also share our vision of possible future challenges and developments in the Internet of Mobile Things.

Type of Paper and Keywords

Regular research paper: *Internet of mobile things, distributed middleware, mobile hub, opportunistic wireless connectivity, complex event processing, bluetooth low energy, IoT marketplace, eessons learned*

1 Introduction

The past decades of research in distributed mobile and pervasive computing have witnessed a significant change. First, the field got a new concept and new challenges: the Internet of Things, which assumes a wider scope than only localized smart environments, handles a much larger number of interconnected nodes/devices (millions or even billion), and where virtually any object of the physical world may interact with other things and humans. Secondly, unlike in “conventional” Distributed Systems, IoT has to cope with a huge variety and heterogeneity of nodes, including embedded devices with very constrained processing and storage capability, limited energy supply, as well as heterogeneous wireless technologies that have much different coverage, connectivity management, data rates, transmission latencies, reliability, and interference robustness, etc. This is worsened by the fact that several WLAN and WPAN technologies operate in the same frequency bands (ISM is 2.4GHz and 5GHz), and thus might potentially interfere with each other. In particular, many smart things use low power wireless technologies and thus have short communication range - just a few meters - such as Bluetooth Low Energy (BLE), ZigBee, NFC, WiFi, LoRa, SIGFOX, WitelessHART, which thus require some other device (hub) that is connected to

This paper is accepted at the *International Workshop on Very Large Internet of Things (VLIoT 2018)* in conjunction with the VLDB 2018 Conference in Rio de Janeiro, Brazil. The proceedings of VLIoT@VLDB 2018 are published in the Open Journal of Internet of Things (OJIOT) as special issue.

the Internet. Thirdly, many of these smart devices will also have actuators, so that they will be able to act on their environment, either individually or in a pre-scheduled/ or coordinated way, and may thus indirectly influence other smart devices. And finally, a very high proportion of these nodes will be mobile, ranging from ordinary smartphone users to sensors or actuators in cars, trucks, cargo packets, parcels, drones, robots, wearables, implants in pets and humans.

Thus, there are many “challenge dimensions” and “tons of problems” that can be addressed when designing and developing a middleware for IoT. In any case, the main purpose is always to provide services and protocols that hide from the application developer the system component’s heterogeneity, distribution, and mobility behind simple and intuitive Application Programming Interfaces (API).

Since in LAC[1] we were always interested in middleware support for mobile communications and context-awareness, we decided to focus on the *Internet of Mobile Things (IoMT)* that subsumes conventional IoT, where usually most smart things are stationary. In IoMT any smart thing, and even part of the communication infrastructure - the hubs at the Edge- may be moved or can move autonomously, and yet remain remotely accessible and controllable from anywhere in the Internet. Therefore, we called these smart things *Mobile Objects (M-OBJs).* Mobile Objects may have very different sizes, movement patterns, movement autonomy, uses and complexity - they may range from terrestrial vehicles of any type (cars, busses, etc.), over mobile domestic or industrial robots, aerial robots (UAVs), to very tiny and light-weight wearable devices, badges or sensor tags. In fact, a M-OBJ may be any movable object that carries sensors and/or actuators and provides some means of wireless connectivity.

1.1 Example of a IoMT Application

The Internet of (Mobile) Things is already having strong impact in several application domains, such as smart cities and homes, environmental monitoring, public security, health care, energy management, asset monitoring, logistics, etc. As an example, consider the delivery of goods or products that require specific ideal transportation and storage conditions on their routes from producer to consumer. For example, meat, fruits, vegetables, or vaccines require ambient temperatures that stay in small ranges (range of 3 to 5 degrees Celsius), or else, special flowers and plants must stay in environments without light and with air humidity above a certain level. By placing some M-OBJs with temperature and humidity sensors close to such goods, and having the sensor values probed regularly, in all stages of transportation and intermediate storage, it is possible to monitor in real time the environment and transportation conditions of these goods along all their transport way. Moreover it is possible to send alerts to the transportation company or the consumer (e.g. the hospital) whenever the safe transportation or storage conditions are starting to be violated. This early alert service can prevent the discarding of such valuable goods and hence the consequent waste of money or endangering of the consumption of spoiled products.

With the goal of supporting development of mobile and IoT applications we started to build a distributed and mobile middleware named ContextNet. This project started exactly seven years ago (2011), initially with little ambition and no idea that it would be later extended to handle IoMT. In this paper, we give a summary of the evolution of the ContextNet, from its birth and the development of early services (Section 2), along its support for discovery and connection with BLE-enabled mobile objects (Section 3), to its current stage, as a micro-service architecture with a rich set of powerful services and tools (Section 3.2) Then, in Section 7 we also present the new research branches in IoMT that we have started in 2017, and the main lessons learned so far (Section 8). Then we shortly present research work that has a similar approach and discuss the main differences to ContextNet (Section 9). We close the paper with the conclusion about the new challenges in managing the code base of this growing project. (Section 10).

2 Genesis and Early Phase

The ContextNet project began in mid 2011 as a contract with the InfoPAE group of TecGraf Institute of PUC-Rio, with the goal to develop a fully decentralized communication software infrastructure to handle geo-location data traffic generated by a large number of trucks, as an alternative to the centralized log-based collection and processing of the InfoPAE system at that time. Based on our previous experience, we decided to use OMG’s Data Distribution Service standard (DDS) [22], with its decentralized P2P architecture and its Real-Time Publish/Subscribe (RTPS) protocol as the basis for inter-node message exchange, but soon realized that it would be unfeasible to have also mobile nodes, i.e. the trucks, as DDS nodes. Therefore, we decided to adopt a two-tier cluster-mobile architecture, where we would use DDS only among the stationary nodes in the cluster or cloud, and use some other IP-based communication protocol to make them interact with the mobile nodes.

[1] Laboratory for Advanced Collaboration of PUC-Rio - www.lac.inf.puc-rio.br

So, in the next six months Lincoln David developed the MR-UDP [26], a light-weight connection-oriented communication protocol based on R-UDP. MR-UDP is a protocol implemented in Java that extends the original protocol by adding mobile node identification orthogonal to the IP-Address, gracefully handling of short-lived wireless disconnections by selective retransmissions, using Protocol Buffers[2] to serialise objects and using mobile-side generated heartbeats to keep MR-UDP connections open behind firewalls. This later feature was essential for enabling connections to mobile devices (mobile phones) of different 3G/4G mobile operators.

Almost during the same time, Rafael Vasconcelos implemented the first version of the Gateway, that was designed to be, on one side, a DDS node, and on the other side, the MR-UDP connection point of mobile nodes with the stationary nodes interconnected through DDS. The main design principle of the Gateway was to be simple, be lightweight, just handle the protocol translation (RTPS to MR-UDP and vice-versa) and publish (to other DDS nodes) connection or disconnection events from any of the mobile nodes connected to it.

Soon after this, Rafael also designed and implemented the *PoA-Manager*, another DDS node that monitors the connection load of all deployed Gateways in a DDS domain and sometimes distributes lists of IP-Addresses of alternative Gateways for connection to each mobile node, that could then spontaneously switch the Gateway (i.e. the Point of Attachment - PoA) with impacting the data and the heartbeat flow of MR-UDP. A mobile node may also be "requested" to change the PoA Gateway by the *PoA-Manager* whenever this one detects an unbalance among the Gateways (mandatory handover). Both kinds of PoA-switching are agnostic to the app executing on the mobile node, because they are handled by the *ClientLib* which also handles all the events and control signals of MR-UDP and exports a quite simple API to the application program. Because we were so confident about our careful and optimized implementation of all these initial components and the well-acknowledged scalability of DDS' P2P architecture, we named it the *Scalable Data Distribution Layer (SDDL).*

Notwithstanding our rather basic development achievements by the end of 2011, we already felt the ambition to design and develop new services and protocols that could facilitate the development of large scale mobile pervasive systems in which location and other context data (i.e. sensor data) should be collected and processed by nodes in a cluster or cloud. This envisioned architecture made of several functional layers was presented in 2011 as a poster in the Middleware conference [11].

[2] https://developers.google.com/protocol-buffers/docs/javatutorial

The first running version of SDDL was showcased in a demo session of SBRC [28]. And to our relief and satisfaction, worked very well, even in the "inhospitable environment" of conference WiFi APs and in front of the gaze of interested students and professors.

2.1 Tests and Extensions

For subsequent publications [27] we then did several performance tests using simulated mobile nodes flooding a simple SDDL core configuration. These showed that SDDL with just two Gateways supported well the mobile tracking communication and management of several (10^{2-3}) thousands mobile nodes, each node producing a geolocation every 30 seconds.

During 2012-2013 we extended ContextNet with additional SDDL-based communication services, such as:

The **GroupDefiner (GrD)** [32] is used to define groups of mobile nodes according to some data sent to SDDL core, such as its context information (e.g. its current geolocation), and then to allow to send a group message to all group members. The GrD is generic in that it accepts *group-selection* plugins, where each plugin defines the processing functions to map a node's context data to a grouID. Hence, each application can define its specific way to tell when some mobile node are in a group. For example, a common use is to set the vertices of a geographic area - for example the limits of a town - and define the group of users that are "within this town". With this, it is then easy to route a same message to all group members just by specifying GroupID in the DDS message, as the Gateways are kept updated of which node belongs to which group.

The **Mobile Temporary Disconnection (MTD)** service is yet another SDDL Core service that aimed at storage and replay of messages directed to mobile nodes, and that could not be delivered to it because of a temporary disconnection. Thus, whenever a Gateway announces that a mobile node is not responding, MTD will hear this announcement and start hoarding the messages addressed to the unreachable node. Then, at a later point, the same node may reappear and connect to a new Gateway, that will then announce that the node is available again. The MTD also hears this announcement, and starts to replay the stored messages. In order to avoid overflow its memory, of course, MTD does garbage collection of the messages, whose policy has to be provided by the application developer, in a similar way than the GroupDefiner plugin.

2.2 SDDL Services

Until the end of 2013 ContextNet regarded only phones and tablets as possible smart things, and it consisted only of SDDL Core services and the ClientLib. Figure 1 shows the SDDL core components - Gateway, PoA-Manager, GroupDefiner and Controller (this latter is the interface to web browsers), all running in a cluster/cloud, and the Android -based software: the ClientLib, as driver and wrapper of the client-side MR-UDP. All components of the SDDL Core use a Pub/Sub interface provided by the Universal DDS Interface (UDI) to interact with each other. The UDI exports an uniform Pub/Sub API similar to the one of DDS, but which hides the idiosyncrasies of the specific DDS product being used, such as OpenSplice, Open DDS, RTI Connext DDS, etc.

At dawn of 2013 ContextNet was then extended with Data Stream Processing capability aimed at the real-time analysis of the streams of context/sensor data produced by the mobile nodes. And among the many existing stream processing systems and languages, e.g. Spark, Flink, Storm-1, StreamIt, etc. we adopted Complex Event Processing (CEP) [20], and in particular the Esper system[3] due to its high expressiveness and flexibility for describing patterns of events, and the ability to build higher level (complex) events from the simpler events of an identified pattern. With the goal of supporting parallel and scalable CEP in ContextNet, Gustavo Baptista, designed and developed the Dynamic Distributed Data-centric CEP (D3CEP) service [2] environment for easy deployment of Event Processing Networks (EPNs) on Processing nodes of the SDDL core, that defined a set of mutually dependent Event Processing Agents cooperating in a complex event detection task.

Also in 2013, Marcos Roriz, that had recently joined LAC, revisited the MR-UDP communication protocol. He fixed some minor bugs, improved and optimized the handling of concurrent client requests, and added the *protocolbuffers* format into MR-UDP, to enable the interoperability between clients written in different languages [25]. This was necessary, as at the same time we developed a Lua[4] version of MR-UDP and ClientLib, already aiming at mobile embedded systems.

3 Embracing IoMT

When Márcio Maia, from the Federal Univerity of Ceará, who was also working on IoT middleware visited our lab in early 2014, Luis Eduardo Talavera Rios, Márcio and us started to discuss about how our middleware could be extended for IoT. So far, ContextNet allowed only to probe the embedded sensors of smartphones, but we knew that for future IoT applications we would need to connect with sensors and actuators embedded in the environment and everyday objects, and that many of them would have only a low power wireless (LoP-WPAN) interfaces as they run on batteries. We then identified that a promising and fast spreading LoP-WPAN was Bluetooth Low Energy (BLE), that had been specially designed for IoT, and that it would soon become standard feature in most smartphones.

All these facts made us consider that smartphones (with turned-on BLE) would be very affordable and convenient communication intermediates between BLE-enabled smart objects and data analytics services running in the SDDL core. And by supporting unrestricted mobility of the things (i.e. Mobile Objects) and the hubs, we would be able to address a yet unexplored set of IoMT applications of three sorts:

- applications where the Mobile Hub is fixed and the M-OBJ is moving (e.g tracking of packets and goods);
- those where M-OBJs are stationary and the Mobile Hub is passing by, (e.g. in participatory sensing where users contribute to the collection of ambient data) or
- applications where both the smart objects and the hub are in movement, and the Mobile-Hub is constantly relaying sensor data about the M-OBJs while both close together, in *co-movement* [29] (e.g. the smartphones of passengers in a bus are connected to a BLE beacon or sensor and send data about trip, for example, which is the temperature in the bus).

Of course, our decision to go with the smartphone, as the Mobile Hub, was also driven by our desire to use the well-tested and efficient SDDL services and the MR-UDP as the "backbone" of a scalable IoT infrastructure, and we already had the ClientLib and some experience with energy-saving geo-location data probing. Although our initial focus was on implementing LoP-WPAN support for BLE to access M-OBJs, we envisioned that in future the M-Hub might also be extended to other LoP-WPAN. This made Luiz Talavera pore a lot to design S2PA, an uniform, yet flexible service for interacting with the M-OBJs.

3.1 S2PA

The *Short-range Sensing, Presence & Actuation (S2PA)* API was designed to be a protocol for short-range communication with M-OBJs, which possess an interface that can be directly mapped to the capabilities of the supported short-range wireless communication

[3] Esper Tech, http://www.espertech.com/esper/
[4] www.lua.org

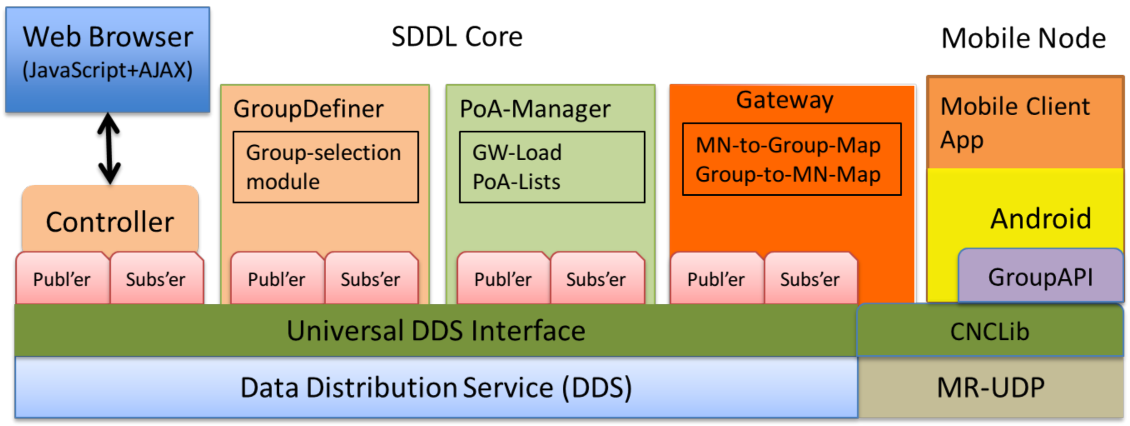

Figure 1: SDDL components Pre-IoMT

technologies (WPAN). To this end, it defines some basic methods and interfaces that all these technologies should implement:

- Discovery of, and connection with M-OBJs;
- Discovery of services provided by each M-OBJ;
- Read and write of service attributes (e.g., sensor values, and actuator commands);
- Notifications about disconnection of M-OBJs.

For this, S2PA defines the *Technology Interface*, shown in Figure 2. The Technology interface includes an ID, defined at programming time, to uniquely identify each technology (e.g. BLE, ANT+, Classic Bluetooth, etc), and a set of required methods that are sufficient for handling a variety of short-range protocols. For example, methods **readSensorValue(),** and **writeSensorValue(),** request a read or write of a sensor, respectively, and **serviceName** represents the sensor name (e.g., "Temperature", "Humidity"). All relevant information regarding M-OBJ's discovery, connectivity, and sensor values obtained from the specific WPAN technology is captured through the **TechnologyListener** which is implemented by the S2PA service, and is either cached or directly forwarded to the SDDL Core.

In its first version we implemented S2PA for BLE and for Classic Bluetooth. Classic Bluetooth was implemented because of the wide spectrum of peripheral devices that use this WPAN technology, and because it is the only means by which M-Hubs can interact directly with each other (without employing Gateways) for handing over discovered nearby M-OBJs. In a later addition, the students of LSDi/UFMA added a new *Technology Interface*, now for the sensors embedded into the smartphones, By this, both device local and device remote sensor data can be probed and processed in a uniform way.

3.2 Adding new Services to the Mobile Hub

By the end of 2014 we then finished and included into the Mobile Hub also the Mobile EPA (M-EPA) service. This service holds a full-fledged CEP engine (Asper, a port of Esper to Android), and thus allows to load, discard, activate and de-activate EPL rules in the Mobile Hub, so that sensor data from the M-OBJs and delivered by S2PA could be promptly analyzed and pre-processed by the CEP engine. This functionality is of great advantage when the IoMT application needs to process data at the edges so that may substantially reduce the data traffic over the wireless link (WiFi/3G/4G) towards the cloud/cluster-based backend services, and also already detect interesting patterns of local events and convey only higher-level events (instead of the simpler data probes) to the backend data analytics services. In March 2015 we officially presented - and ran a demo - of the Mobile Hub in a IEEE PerCom workshop and demo track [29].

Then, in 2015, Luiz E. Talavera did another very important re-engineering on the Mobile Hub: he introduced the EventBus Publish/Subscribe[5] for local, intra-Android communication, instead of the inefficient Intents and Broadcasts of pre-Android 4.0. This turned the Mobile Hub architecture into a truly micro-services architecture, where several services (except the Connection Service and the S2PA) can be deployed or not, depending on the requirements of the IoT Application using the Mobile Hubs.

[5] http://greenrobot.org/eventbus/

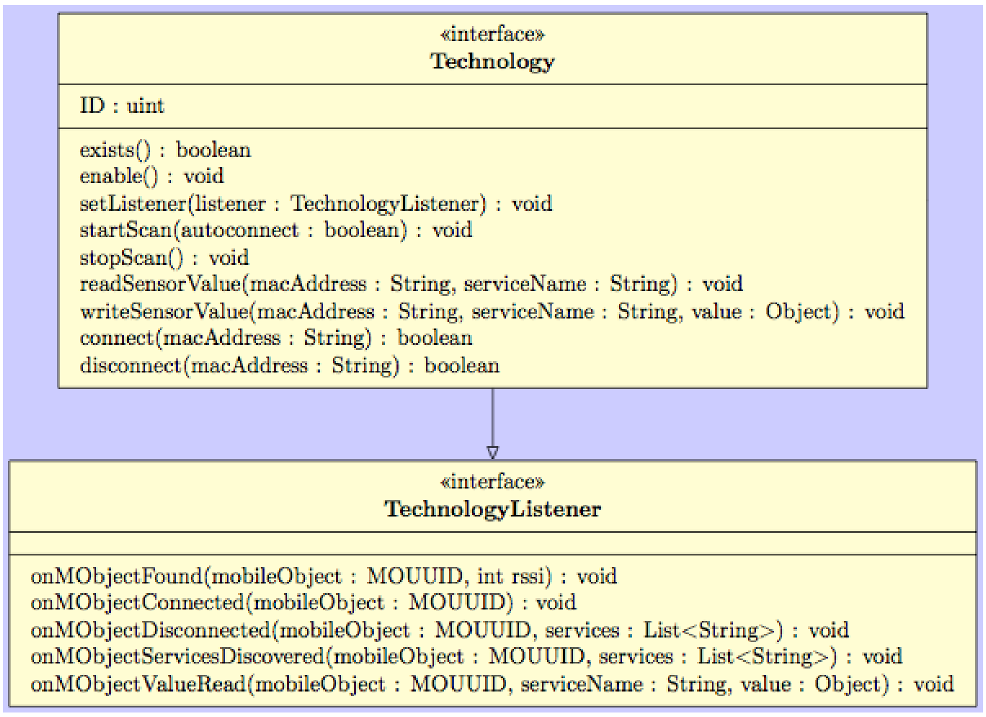
«interface»
Technology
ID : uint
exists() : boolean
enable() : void
setListener(listener : TechnologyListener) : void
startScan(autoconnect : boolean) : void
stopScan() : void
readSensorValue(macAddress : String, serviceName : String) : void
writeSensorValue(macAddress : String, serviceName : String, value : Object) : void
connect(macAddress : String) : boolean
disconnect(macAddress : String) : boolean
«interface»
TechnologyListener
onMObjectFound(mobileObject : MOUUID, int rssi) : void
onMObjectConnected(mobileObject : MOUUID) : void
onMObjectDisconnected(mobileObject : MOUUID, services : List<String>) : void
onMObjectServicesDiscovered(mobileObject : MOUUID, services : List<String>) : void
onMObjectValueRead(mobileObject : MOUUID, serviceName : String, value : Object) : void

Figure 2: Main two interfaces of the S2PA

3.3 Current Mobile Hub Components

The M-Hub is multi-threaded and consists of the following Android services and managers, all executing in background, i.e. independent of the user apps. Figure 3 depicts these components. The **LocationService** is responsible for sampling the M-Hub's current position and attaching it to whatever message is sent to the Gateway (GW), which can be either a static, manually entered geo-point, or the latest geo-coordinate obtained from the smart phone's embedded GPS sensor. The **S2PA Service** implements the TechnologyListener and interacts with all nearby M-OBJs that "talk" the supported WPAN technologies. This service is responsible of the discovery, monitoring and registration of nearby M-OBJs, by periodically doing scans for each supported WPAN. Depending on the kind of interaction (and the WPAN technology capabilities) a communication link may be established with some M-OBJ, over which the M-Hub will interact in a request-reply mode. Data packets and messages from/to M-OBJs may have different formats and encodings, so it will also transcode sensor data and commands from the specific M-OBJ-specific data format to serialized Java objects, for transmission to the GW, and vice versa. Internet messages are received from - and sent to - the Gateway by the **ConnectionService**, which runs the ClientLib for communication with the SDDL Core and, in order to optimize communication over the Internet link, the M-Hub may group several pieces of sensor data or commands assembled by the S2PA Service into a single "bulk message" for transmission. It is also important to mention that some messages (e.g. M-OBJ connection/ disconnection) have a *high delivery priority* so that they will be relayed directly to the SDDL core, instead of being buffered for further bulk Internet transmission. The periodicity and duration of all of these three services' actions, is influenced by the device's current energy level (LOW, MEDIUM, HIGH). This will be set by the **Energy Manager**, which from time to time samples the device's battery level and checks if it is connected to a power source.

3.4 Further Extensions

The Mobile-Hub is evolving continuously as new functionalities are demanded. In 2015 we started to investigate the support for quality parameters related to the context data collected from M-OBJs and the distribution service. The term QoC (Quality of Context) has been usually defined as the set of parameters that express quality requirements and properties for context data (e.g., precision, freshness, trustworthiness, etc.) [6]. In the last years, much research has recognized the approach of introducing the quality of the context data distribution (e.g., data delivery time, reliability, etc.) in

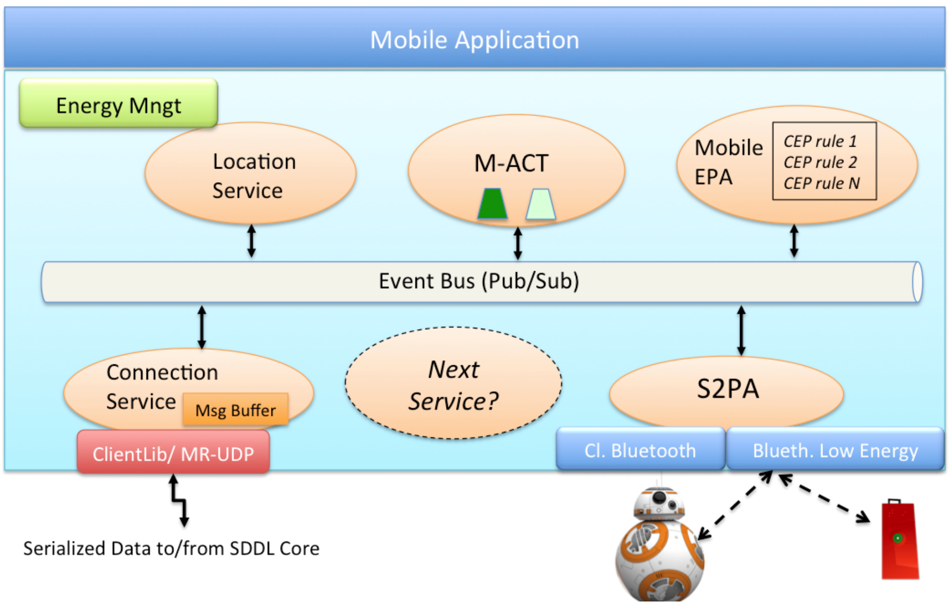

Figure 3: Current mobile-hub architecture with services interacting with two M-OBJs through BLE and classic bluetooth

order to ensure the availability of the context data with the right quality, in the right place, and at the right time. In this broader view, QoC has to consider the quality of both the exchanged context data and the distribution process to ensure user satisfaction [4].

The quality of context has a significant impact on the behavior of context-aware applications and the efficiency of the services offered and can greatly influence the user experience [7]. Therefore, satisfying the requirements of QoC for IoT/IoMT applications is a very important step to ensure a correct execution of the applications and the satisfaction of their users. As an example, one can consider a remote patient monitoring application, where it is crucial to ensure the required data precision, freshness, reliability and data delivery time for correct application execution.

In 2015 Berto de Tácio started the investigation of how to provide a comprehensive QoC support for ContextNet. He developed CDDL [15], a Context Data Distribution Layer at the top of M-Hub. The proposed solution combines a mobile gateway (the M-Hub) for the acquisition of raw data from heterogeneous physical sensors with the CDDL, responsible for registering and discovering the available context services, as well as for provisioning and monitoring context information and for ensuring the context data and distribution service quality.

The CDDL provides an extensive support for both, quality of data (QoI) and quality of service (QoS) parameters. Concerning QoI, the available parameters are: Accuracy, Source Location, Measurement Time, Arrival Time, Expiration Time, Age, Measurement Interval, Available Attributes, Completeness, and Numeric Resolution. In respect to QoS, it provides: Deadline, Refresh Rate, Latency Budget, History, Destination Order, Lifespan, Retention, Vivacity, Reliability, and Session.

The CDDL also provides a M-OBJs discovery service, allowing the applications to issue two types of discovery queries: instantaneous and continuous. The instantaneous query returns the available service providers that meet a given criteria at that moment the query was issued. The continuous query not only returns the service providers meeting the specified criteria at the time it was issued, but also instantiates the query in the Monitor component, which causes the middleware to continuously evaluate the query as new services providers are discovered. This latter type of discovery query is particularly useful in the mobility scenarios that characterize IoMT. As part of the discovery criteria, applications can request service providers that meet specific QoC requirements, such as the ones providing a given accuracy.

Since several QoC parameters exhibit dynamic variability (they oscillate over time), the CDDL provides a Monitor component for analyzing context data streams in order to detect the occurrence of certain events that are of interest to the application, such as a variation of a given QoC parameter (e.g. accuracy). CDDL also offers a Filter component, that filters information based on the content of its attributes, including the QoI metadata.

4 HORYS

In 2017 we realized that recording the encounters between smart M-OBJs and Mobile Hubs could be the cornerstone for many IoT applications what need to track persons, vehicles or machines. For example, tracking packets and carts for logistics, employees and assets in industries, patients and health professionals in hospitals, etc. In all cases, either BLE sensors/beacons are carried around, with (not so quite Mobile)-Hubs executing in RaspberryPi boards or other computer boards being attached to rooms or halls, or the other way round, Mobile Hubs being carried and detecting beacons in each relevant place.

This use of IoMT required a middleware service that is able to collect and store a large volume/stream of these encounters (*Rendezvous*, for IoMT applications with many mobile entities. HORYS (Hub-Object-Rendezvous RegistrY Service) is thus ContextNet's registry service responsible for storing and querying Rendezvous events. It provides several options for querying which M-OBJs a certain M-Hub met, or which M-Hubs received a beacon from a particular M-OBJ. Moreover it allows to query and classify Rendezvous events by location, by WPAN RF signal quality, and elapsed time of the encounter. HORYS executes at a SDDL Processing Node and uses the NoSQL MongoDB technology to store the Rendezvous data and perform highly optimized and parallel searches on this data store.

However, HORYS is completely generic and agnostic to the semantics of the holder or place with M-OBJs and H-Hubs, and thus can be used for many IoMT applications. It is only focused on very fast, parallel, data insertion and retrieval of encounters, and how to do this for several Gigabytes/second. On the other hand, HORYS does not associate semantics to Rendezvous events, for instance, it does not know that a beacon B is assigned to a particular patient rather than to a nurse. This association is made by the Hospital 4.0 Semantics service, which maps the beacons to specific users. Furthermore, it also maps the location of such events to the hospital rooms and facilities. HORYS has been developed by Marcos Roriz Jr (in mid 2017) and is the core service of the Hospital 4.0 data analytics application.

5 IoTrade

Similar to current commodity trade markets, where buyers don't need to know - and directly interact with - sellers/producers we believe that something similar may also happen for IoT services (data, actuators, connectivity and analytics). Instead of engaging in a direct service contract the IoT client, a user interested in information about - or the ability to actuate upon- things spread our in the world, might just want specify some required attributes about the needed sensors, actuators or expected internet connectivity. Then all providers of such services satisfying the required specification would be able to offer and sell their service. Hence, in a possible future *IoT Marketplace*, sensor data and IoT services in general will be classified according to their location, their level of precision, freshness, latency, scope, trustfulness, availability, and other attributes.

Along this vision, IoTrade is ContextNet's brokerage service implementing an IoT Marketplace [23]. It discovers the properties of existing sensors, actuators, connectivity providers (e.g. owners of a smartphone with Mobile Hub) and analytics services, performs continuous quality verifications and classifications of these elements and services, as well as the matchmaking between smart object (sensor/actuator) , connectivity access and data analytics providers, on the one hand, and IoT application clients with specific demands on the other hand. The IoTrade consists of a mobile client application, which is the interface for setting the IoT application client requirements and a server component, executing in the SDDL core where the matchmaking algorithm is implemented.

The matchmaking algorithm aims at selecting a combination of providers (smart objects/sensor, connectivity & data analytics) that best fits the IoT application requirements input by the client. These requirements include the amount the customer is willing to pay, the minimum required QoS, and the user's current location, since we implicitly assume that the client seeks access to sensor/actuators which are in his/her vicinity. The client cannot choose which exact provider will provide the service, the algorithm alongside the data commoditization is going to choose the combination based on QoS parameters. In addition, IoTrade keeps continuously checking the quality of all current resources and automatically swaps resource providers if it detects a disconnection or a drop in the quality of the offered resource/service.

6 Implemented Applications

Over the past years, several prototype applications using ContextNet were developed by students as part of their Masters or Doctoral thesis, or as a class project. The early ones just used the mobile communication and group communication features of ContextNet, while the more recent ones are already IoMT applications.

Bus fleet tracking and communication: In 2012/2013 a bus fleet tracking and IM communication application (Aplicações de Rastreamento de Frotas e

Fiscais - ARFF) [31] was developed as part of a class project. It included a dashboard - for the Highway Control Central - with a map displaying the locations of all buses and inspectors, the ability to dispatch inspectors, and to follow on-line how the inspector is filling in the check-up form.

UAV swarm coordination: A second major application was the swarm coordination protocol for mobile flying robots developed in 2014 as part of Bruno Olivieri's Master thesis [10]. It used ContextNet's group communication to distribute the robot leader's steering commands and position reliably and timely to the remaining flying robots of the swarm. Since the main focus of the research was to identify the necessary wireless latency requirements so that such group steering would work properly without causing much error in the robot's relative positions, we did not actually build and piggyback smartphone with the Mobile Hub on each UAV, but only simulated and showed the animation the collective swarm control on a map[6].

Detection of reckless driving: Igor Vasconcelos did research on correlating data from smartphone sensors and data from the on-board sensors of cars to identify, in real-time, reckless/dangerous driving behavior or drivers. And since this data analysis is quite data intensive and has to be repeated at high frequency, it was suitable to do it directly on the smartphone, while only sending eventually the outcomes to a server for sharing this information with other stakeholders (e.g. an insurance company, or the driver's relatives). This application was implemented using Mobile Hub's S2PA service to connect with the OBDII toggle, wirelessly receive data from the vehicle (e.g. speed and RPM, etc.) and probe other data from the smartphone's embedded sensors (e.g. accelerometer) [17]. He also used M-EPA, for a CEP processing of the combined sensor data streams. The main contribution, though, was to convert several batch outlier detection algorithms found in literature to stream processing ones, and describe them as CEP event pattern detection rules.

Pervasive RPG game: Since 2017 Pedro Igor has been developing a mobile RolePlayingGame (RPG) that uses BLE sensors and beacons spread across the physical spaces and aims to enhance the gameplay through real-world presence and interactions. Because it was primarily conceived for use at the PUC campus - and inspired by Pokemon Go - it was named *PUCmon*.

The implementation of this pervasive game consists of a mobile client - that runs the game client app in foreground, and the Mobile-Hub in background - and also a Game Server, that runs on a SDDL core processing node. While de former is responsible for discovering and

[6] Video on Youtube, URL https://youtu.be/3phH-5e7l9c

Figure 4: Screens of the pervasive RPG game

connecting with BLE sensors/beacons placed near to the smartphone of the user (the gamer), the latter resolves the conflicts of multi-player resource access, and also manages the gameplay-specific information associated to each of these beacons or sensors. For example, when a player's smartphone connects to a SensorTag with a temperature sensor, then the actual temperature reading may be used as the basis of calculating and displaying some game-specific item or event on the game screen 4.

This research and game development was motivated by our assumption that pervasive games using IoMT may be used to support participatory sensing and may introduce new forms of entertainment, learning, and socialization.

7 New Research Branches: the Future

In early 2017 we started several new IoMT research branches based on the core components of ContextNet. These new branches, which we called R&D Divisions, tackle several issues and innovative approaches to IoT that so far have apparently not been given much attention by the academic community.

7.1 Generic Actuation Support

Many IoT middleware systems have been developed focused at supporting sensors-to-cloud communication and processing capabilities at the edges, but surprisingly very little has been proposed or implemented towards IoT actuation over WPAN links. Actuators are part of some smart things, and cause changes to occur, such as turning on a motor, or shutting a valve.

Of the few research works that mention actuation on a smart devices it is treated as a simple issue. In commercial IoT systems, however, the actuation-control logic is usually hard-coded and intertwined with the remaining application logic. The main reason being that IoT applications are tailored to very specific - and proprietary - smart things with particular/proprietary actuation protocols and low-level actuation instructions and feedback signals. Furthermore, most of IoT systems assume a stable wireless connection between the wireless gateway and the smart things, which is not the case in the IoMT.

In this R&D Division we are investigating the problems of actuation of M-OBJs over short-range links subject to intermittent connectivity (due to relative mobility of the M-Hub and the M-OBJs) and have proposed and prototyped a ContextNet service and protocol that supports such generic actuation. For this, two new ContextNet components, the Mobile-Actuator (M-Act), a micro-service of the Mobile-Hub (for Android devices), and the Smart Objects Manager (SOM), a micro-service of the SDDL core executing in cloud/cluster were developed.

In order to cope with the heterogeneity of actionable M-OBJs, which can range from simple light bulbs, LEDs or bells to complex machines or vehicles, our approach enables that the IoT client defines actuation controls as sequences of high-level and generic Actuation Control Commands (ACC), which are then translated to corresponding low-level strings of byte-codes (that are actually recognized by the specific M-OBJ's actuation control circuit. In our approach, this ACC-to-bytecode translation is done by a driver that is specific for the kind, make and model of the M-OBJ, and will be previously downloaded, on-demand, into M-ACT. By this uniformity of the ACC language, it will be possible to define actuations that are largely independent of the concrete smart things encountered, which is essential for the success of the opportunistic interaction of M-Hubs with M-OBJs while it roams in through the different ambients. Moreover, it will be possible to write scripts of ACCs defining complex and coordinated actuations between M-OBJs.

With our current implementation of the Generic Actuation Support [30], we have managed to issue ACC commands to control the motion and the LEDs of the mobile toy robot BB-8 from Sphero[7].

7.2 Stream Reasoning

The goal in this R&D Division is to investigate the problems and advantages of providing the capability of a IoT application to reason about the environment, the people and the system, and how this can be supported by distributed middleware services and APIs. The main requirement is that such reasoning and deduction of new information should be performed in real-time, and should be based both on the event stream (raw or processed data stream from sensors) and on Deduction Logic over semantic/ontological model about the physical world. Ontologies provide a means of knowledge representation; they capture a domain of interest by formally defining the relevant concepts in the domain, and the relationships between these concepts.

Hence, we look for a *stream reasoning process* that is able to deduce new events that have not actually been monitored, but which can be inferred from the relationships between concepts and events, described in an ontology. These may be spatial relationships (e.g. if objects O_1 and O_2 are stacked and the lower object O_1 is removed, then O_2 will fall); temporal relationships (e.g. if a tire is rapidly loosing pressure at instant t_1, then in $\mathrm{t}_2 < t_1 + 5$ min the car of the tire will have to stop); or else, contention relationships (e.g. if batteryType$_B$ can ignite, and batteryType$_B$ is a component in all smartphoneModels$_S$, then all items of smartphoneModels$_S$ may become damaged).

After defining the general architecture for stream reasoning [12], we have implemented our first prototype of ContextNet's Stream Reasoning Service (SRS), which utilizes Esper for transforming the stream of raw sensor data into a stream of RDF triples, that are processed by a CSAPQL reasoner, that receives the RDF ontology with the knowledge base. In our preliminary tests, we noticed that the performance of the continuous reasoning process (i.e. its throughput) is much dependent on the size and complexity of the ontology. Therefore, to make feasible real-time inferences, the knowledge base should contain only few instances (e.g a moderately small A-Box), and the model of the physical world should be straight and avoid unnecessary generic concept types.

As next steps, we plan to construct more stream reasoning scenarios, investigate more about which are suitable ontology structures, and how we can decompose the ontology into sub-ontologies, so that reasoning may be performed in a decentralized way, maybe enhancing the power of the CEP processing stage.

7.3 Smart City Support

As a participant of the National Institute of Science and Technology on Smart Cities (INCT InterSCity)[8], we are working towards the integration of ContextNet with the InterSCity platform and a city-wide use of ContextNet at a universal communication infra-structure for connecting mobile phones with wireless sensors and actuators spread in streets, parks, bus stops, city malls, etc. The initial technical challenge is to guarantee scalability and deployment over multiple network domains, and later, also administrative domains (environmental, security, health, traffic, etc.), as well as to handle the interoperability of many different types of

[7] www.sphero.com/starwars/bb8

[8] http://interscity.org/

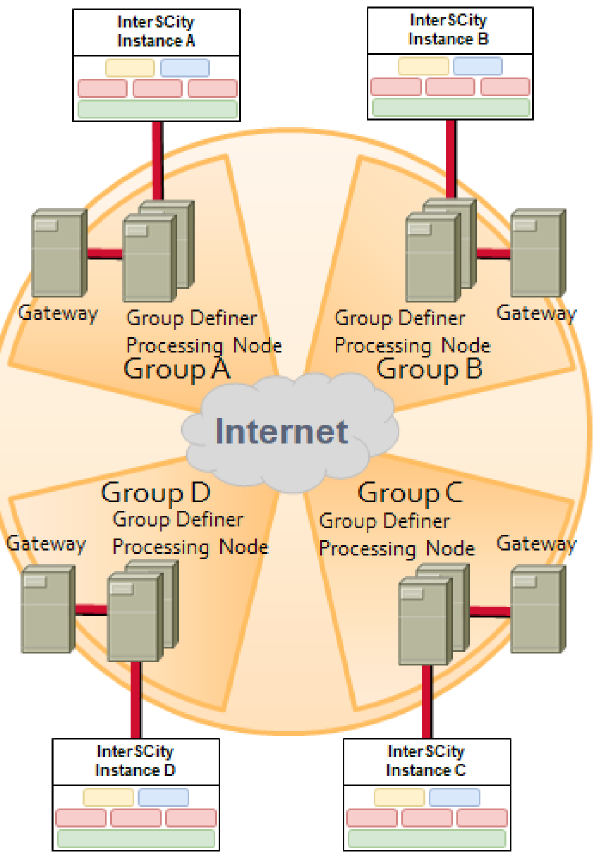

Figure 5: MUSANet: Multi-domain InterSCity-ContextNet pairs

sensors and actuators.

Along this R&D line, we designed MUSANet (Mobile Urban Sensing and Actuation Network), a distributed hierarchical context-aware system for capturing, storing and processing urban sensor data, sending data to actuators, and receiving and publishing information through publish-subscribe protocols. MUSANet uses and integrates the InterSCity [3] platform and the ContextNet IoMT middleware. The InterSCity open-source micro-service platform is used to store in a structured way, define the city resources, and manage resources and sensor information in an efficient way. InterSCity provides the basic blocks for the development of applications related to smart cities through REST APIs. Of ContextNet the SDDL core components are used, specifically the GroupDefiner and the POAManager, and of course, the Mobile Hub.

MUSANet uses various distributed ContextNet sites that are connected through IP tunnels using the Internet infrastructure to create one ContextNet infrastructure. Several network topologies can be used, including star, hierarchical or full-mesh format, with or without path redundancy. Sensor Data is captured using the Mobile-hub - by making participatory sensing campaigns - and are analyzed in real time through Complex Event Processing at the Mobile Hub and D3CEP. In MUSANet approach, the city is divided into groups or regions based on sensor distribution and not just neighborhood or zones. These regions can (and should) have intersections, and there is also the possibility that regions encompass entirely more than one region. Each region must have a set consisting of at least one Gateway, a GroupDefiner, a Processing Node, and an InterSCity instance connected through a local area network, as shown in Figure 5.

7.4 Edge Computing Security Architecture

Security is a very important issue for IoT, and any middleware should implement secure protocols to access smart things [19]. While there are many and well-known means to control and secure the mobile-to-cloud access and communication, there are only few works that tackle the *last meter secure access*, between the WPAN/WLAN gateway (i.e. the Mobile Hub) and the smart mobile thing M-OBJ, in our case, over the BLE link.

Since the Mobile Hub is the intermediate of the communication between the SDDL Core service and the smart mobile things (M-OBJs) it is also the place where credentials must be checked and access must be controlled. And with this in mind, we designed *EdgeSec*, the Edge security architecture for ContextNet [13]. In this architecture, the Mobile Hub has two basic application services to enforce the security of IoT applications: (i) smart thing control service; and (ii) the access control service. The first one operates as a firewall proxy by intermediating the communication between the cloud and the M-OBJs, being able to inspect the protocol messages in order to detect and block malformed ones that could harm the smart things. The access control service aims to offer a robust access control service to validate authentication credentials and restrict access to authorized users.

Another element of EdgeSec architecture is the requirement that the M-OBJs should provide distinct operating modes: (i) configuration mode; and (ii) service mode. The first one allows configuration actions such as the modification of operating parameters (e.g. signal strength, cryptographic keys, network address, authentication method) and updating of the firmware, among others. The latter one is the common operating mode in which the smart thing do what it is intended to do and allows data to be collected. As a security measure, the smart thing shall use an access control method before switching modes, such as validating a PIN (Personal Identification Number).

We are now in the process of implementing the smart thing control service and the access control service into the Mobile Hub. In parallel we are studying Bluetooth LE and looking for the best way to incorporate the configuration mode of M-OBJs into the BLE stack, so that it becomes transparent to the code on the M-OBJ. After this, we will develop a toy distributed application example to show the end-to-end interaction between a M-OBJ and a remote client that will consume sensor data and is able to configure M-OBjs.

8 Lessons learned

Sometimes it is good to take a step back, look at a long process - in our case, 7 years of R&D (cf. Figure 6) - from a broader perspective, and try to distill what we have learned. After some reflection, we identified that following technical and organizational issues contributed to the success of ContextNet.

Micro-service architecture: Since IoT has so many and diverse applications, it is clearly impossible to predict which middleware services and protocols will be required in the next few years or the next case study. Therefore it is very important to design from the beginning a flexible and extensible software architecture constituted of independent services that interact with each other in loosely-coupled way, preferably through an asynchronous communication mechanism (e.g. an event bus, tuple space, or Publish/Subscribe). In ContextNet this is done by the DDS Pub/Sup communication, and in the Mobile Hub by the EventBus, allowing to plug-and-play with micro services so as to satisfy the needs of the IoMT application.

Careful choice of communication technologies: Choose the underlying communication protocols and technologies based on their suitability to IoT traffic, their general adoption, and the expected market penetration . In ContextNet we bet on that a connectionless protocol over IP is preferable for mobile nodes, we used DDS as it is a well-proven standard for scalable communication with many QoS parameters (but only effective within a LAN or datacenter/cluster communication matrix), and on BLE, due to its efficiency and its low power consumption.

Scalability first: From the early phases of design we considered scalability as a non-negotiable requirement, and developed ContextNet?s services and protocols accordingly. Among the many possible types of scalability, we focused on the capacity of handling large numbers of Mobile Hubs and their interaction through SDDL core, since indirectly these determine the number of smart mobile objects that can be accessed. Although so far we have not run any ultra-large test of ContextNet with millions of mobile objects, we have already shown that each Gateway can handle well up to 10.000 simultaneous connections to (simulated) Mobile Hubs. And since each Gateway can run on a different machine with a public IP address in the cloud/cluster, and because DDS can **handle up to thousands of static nodes** we have, in principle, thousands of Gateways serving a million-and-more Mobile Hubs. But the ability to have so many Gateways working together will largely depend on the communication infrastructure within the SDDL Core, and of course, on the data traffic from and to the Mobile Hubs. According to the BLE 4.2 specification, it is possible to keep active up to 2000 connections between the Mobile Hub and the Mobile Objects. However, this will largely depend on the specific mobile platform. Hence, we learned that a properly designed decentralised architecture plus a careful choice of the communication technologies are fundamental for being laying the grounds for

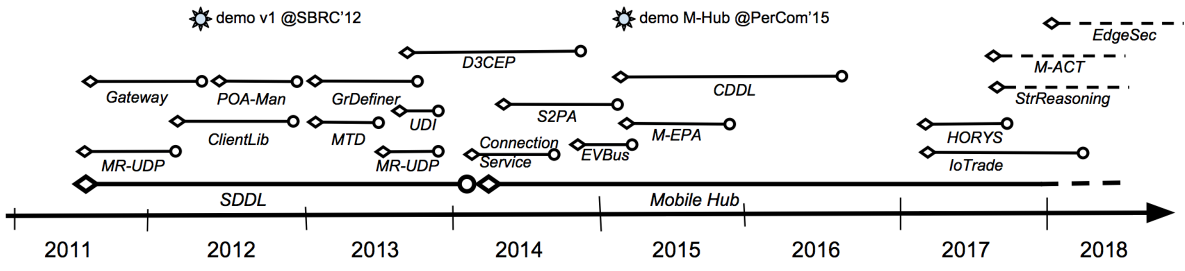

Figure 6: ContextNet evolution over time

a scalable system, but that the actual maximum number of supported smart objects will depend on the actual software products being used and on the application data traffic.

Mobility second: In the same way as mobility is intrinsic part of live and work of humans, it will also be the main characteristic of smart objects, tools, machines, sensors and of course vehicles, which together constitute the IoMT. Since mobility, intermittent (wireless) connectivity and handovers are central issues in IoMT applications, all designs and implementations in ContextNet took these as the central challenges to be considered, and always tried to support as best as possible these disruptive events. This translated, for example, into the application-agnostic reconnection/ handover of Mobile Hubs between Gateways, when their WWAN connection breaks, or the MTD service which buffers SDDL-outbound messages when a Mobile-Hub is temporarily disconnected. Another example is the option for BLE as the main WPAN technology, which supports quick discovery of, and connection with peripheral devices (with a handful of services, and assuming a 1Hz high-frequency scan), in less than 2 seconds. Or else, the *command replay* mechanism in M-ACT when mobile smart device with actuator has not stayed connected to Mobile for sufficiently long time.

KISS - Keep it small and simple[9]: For a middleware to satisfy its clients - the application developers - of course, it has to be stable, extensible, reliable, scalable, easily configurable, etc. But in order to attract new IoT developers, it has to have three main properties: it must be easily installable, it should have a good online documentation and should provide simple and intuitive APIs. While the first two properties are fairly obvious, the issue of simple APIs is not. We learned that ContextNet was well accepted by new developers because it exposed only few concepts and a small API with few parameters and options. For example, ClientLib provides just "connections" (i.e. events *established connection/broken connection*), all nodes and groups have essentially the same UUIDs, and also UDI exports just a small set of the most utilized DDS Publish/Subscribe primitives. With this, most development needs are satisfied, and the user is shielded from the complex and nasty details of processing and handshaking can be ignored. Of course, this has the disadvantage of less freedom to configure the system according to the particular needs. But as ContextNet is still "under construction", we happily accept these requests and do our best to include it in the API of the next version.

[9] A variant of the well-known? Keep it Simple, Stupid? (KISS principle)

Foster Dev community spirit: A software can only evolve and improve if it there is an active community working and using it. Fortunately, since its beginning in 2011 ContextNet has always been maintained by on a group of very helpful, cooperative and very experienced programmers, who promptly fixed bugs and interacted with desperate local and remote students. But as it is well known, the main drawback of academic software development is the quick turnaround time of developers, which usually leave after they graduate. Therefore, more than in other ITC business it is imperative to cultivate the dev community spirit around the software system, and keep the former developers engaged in helping and giving (remote) support to the novice developers. Of course, it is important to train also some local students about the entire system, so that they can act as the "local wizards" of the system. By this, each developer stays in touch with the group, feels proud for his/her contribution and also acts as an evangelist for the project and the philosophy behind it. Fortunately, and without being aware that this was in course, we have managed to create a ContextNet community which now spans several research groups in universities in Brazil and abroad.

Keep open for new technologies and approaches: In the same way as the ContextNet project was expanded towards IoT in 2014 by supporting Bluetooth, it may happen that in the next years it may incorporate a distributed ledger implementation, WearOS, or a new wireless technology for IoT like NB-IoT. Therefore, one should never regard a software system as a compact, closed product that need only be maintained, but instead always consider and prototype new features into the software base, keeping it a live entity, where some parts evolve into well established and polished services because they are felt necessary by most users, while other parts remain only small "stubs of past experiments". However, these free experiments with a software and the entailed "wasted" efforts in dev work time may not be feasible in the corporate world. But in academic research this is not only allowed but even expected, as the main goal here is to innovate, test out ideas, make experiments and train students. And in terms of the product itself, the software system, it will usually be designed to be simple and easy to change/incorporate new technologies. In ContextNet, we can see this several services of SDDL Core and in Mobile Hub's services S2PA and M-ACT.

9 Related Work

There are many approaches to middleware for IoT [24, 21]. Some of them concentrate more effort in specific challenges, such as security ([9]) and interoperability ([8]), others are focused in specific domains, such as Smart Cities applications ([5]), while others provide a more comprehensive support for IoT application development. However, many of them don?t consider mobile nodes, do not consider movable smart objects, or do not scale. We are unaware of a systematic approach and scalable middleware architecture focused on the Internet of Mobile Things, in which the connectable things can be moved or can move independently, and are accessible and controllable from anywhere intermittently.

The first reference of use of smartphones as IoT gateways was a position paper by Golchay et al. [14]. But as expected, their software architecture of the gateway is rather high-level, uses traditional protocols (e.g., TCP, UDP), and does not mention any concrete short-range, low-power WPAN or WLAN technology.

A much more concrete design of a mobile gateway (running on smartphones) is the work described in Aloi et al. [1]. The software architecture supports opportunistic discovery, control, and management of IoT devices, along with data processing, data collection and dissemination capabilities on a continuous basis. In addition, it can send control messages or data streams, such as streaming video, to neighboring IoT devices opportunistically. The architecture presents a multi-standard, multi-interface and multi-technology communication structure capable of integrating different communication standards and radio interfaces that presents a reduced use of hardware resources. The flexibility of the framework presented is guaranteed by the modular implementation that allows the possibility of extending the framework by adding new services (NFC, BLE, etc.), if necessary. One limitation pointed out by the authors is the high power consumption, mainly due to the simultaneously active radio interfaces combined with the small battery power of smartphones, which limits the smartphone lifetime. The authors, however, claim that their approach is still feasible as technological advances related to battery issues and radio interfaces will, in the short term, make these problems irrelevant. The solution is being used in real cases of IoT applications (Smart Health of the INTER-IoT project and Smart Street of the Res-Novae project).

Although the general approach presented in Aloi et al. is similar to ContextNet's one, the M-Hub has some distinguished features: it overcomes the power consumption problem through dynamic adaption of its functionalities based on the available battery level and also by allowing the selective activation of network technologies and individual sensors. M-Hub also provides the support for local (in-network) data processing through the use of application deployed CEP rules or Java code. When combined with CDDL, ContextNet provides an extensive support for QoC parameters and also allows continuous and instantaneous discovery of M-OBJs services, monitoring and filtering of context data.

Moreover, He et al. [16] describe MODE, a middleware that can dynamically change its deployment of function modules based on context awareness to adapt to environment changes. The middleware architecture is based on two layers. The Device Node Layer is responsible for collecting, cleaning and aggregating the data produced by the sensors, transmitting this data to the Server Layer through the MQTT data distribution protocol. The Server Layer is responsible for context science. For this, Complex Event Processing (CEP) is used. This layer is also responsible for managing the execution of tasks at run time, which are done through logical scripts. In this way, it is possible to dynamically perform tasks based on context awareness to adapt to changes in the environment. MODE provides developers with a set of basic tasks to handle a large number of scenarios, however, it is also allowed the developer to customize a task, thus developing an extension of the basic task to meet their specific requirement.

In ContextNet, the end user application can be either local (entirely executing at the smartphone with the M-Hub) or distributed (at the M-Hub, in the SDDL core and/or any other end user device), while in MODE applications are always remote Web applications. The M-Hub, as in the MODE Device Node Layer, is responsible for collecting data and also supports the execution of filtering and aggregating functions. A major difference is that M-Hub allows the dynamic deployment of either CEP rules or Java code for performing any user defined processing, leading to a more flexible approach for in-network, edge processing. Through CDDL, ContextNet provides support for QoC management, filtering and monitoring, issues that are not addressed in MODE. Furthermore, ContextNet also support CEP processing in the SDDL Cloud through D3CEP, similar to what is provided in the MODE Server Layer. However, ContextNet does not provide a scripting language for defining tasks that can be triggered based on context data, as the one provided in MODE.

10 Conclusion

After MoCA [18], ContextNet has been our second experience of building a middleware system developed by several “generations of graduate students”. ContextNet has been used primarily for many kinds of research and prototyping of new middleware protocols and services considering mobility as a premise. Therefore, over the past four years it became a large assortment of quite different - and sometimes incompatible - extra middleware services and IoMT applications, developed at LAC in PUC-Rio and LSDi in UFMA. Nevertheless, the main pillars, the SDDL Core and the Mobile Hub mirco-service architecture, have remained unmodified and served as a reference for all other developments.

But as the project now is spreading its influence to, and gaining new collaborators from, several other research groups in Brazilian universities (e.g. Federal Fluminense, Federal of Goiás, Federal da Paraíba, Estadual do Ceará) and even abroad (e.g. LiP6, U. Kaiserslautern), we are now creating better communication channels among the researchers and developers and trying to improve even more the system’s documentation[10]. Moreover, we will need to decide which added services are to be incorporated into the main “product version” of ContextNet, so to ensure that all other users get access to well documented, stable and reliable services.

In any case, the ContextNet project has provided very interesting scientific, development and organizational challenges. It has been - and hopefully will continue being - great fun to develop and manage its evolution.

[10] http://www.lac.inf.puc-rio.br/dokuwiki

Acknowledgements

ContextNet is collaborative work of following people (in order of appearance on “the stage”): Gustavo L. Baptista, Lincoln D.S. Nery, Rafael O. Vasconcelos, Rafael André, Igor O. Vasconcelos, Luciano Sampaio, Caio Seguin, Lucas Alves, Bruno Olivieri de Souza, Davi Rocha, Hisham Muhammad, Marcos Roriz Jr, Luis Eduardo Talavera Rios, Felipe O. Carvalho, Vitor Pinheiro, Ruhan dos Reis, Luiz Pitta, Sheriton Valim, Matheus Zeitune, Berto T.P. Gomes, Luis Carlos M. Muniz, Alexandre Meslin. **Thank you for your hard work and enduring enthusiasm!**

We also greatly acknowledge the TecGraf Institute of PUC-Rio for its continuos financial support during these seven years. Thanks go also to Microsoft Brazil, FairCom Inc., Intel Brasil and to CNPq for donations, student and faculty scholarships and to the INCT Future Internet for Smart Cities, financed by CNPq (465446/2014-0), CAPES (88887.136422/2017), and FAPESP (2014/50937-1).

References

[1] G. Aloi, G. Caliciuri, G. Fortino, R. Gravina, P. Pace, W. Russo, and C. Savaglio, “Enabling iot interoperability through opportunistic smartphone-basedmobile gateways,” *Journal of Network and Computer Applications*, 2016.

[2] G. Baptista, F. Carvalho, S. Colcher, and M. Endler, “A middleware for data-centric and dynamic distributed complex event processing for iot real-time analytics in the cloud,” in *34th Brazilian Symp. on Computer Networks and Distributed Systems (SBRC 2016)*, May 2016.

[3] D. Batista, A. Goldman, R. Hirata, F. Kon, F. 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*. IEEE, 2016, pp. 1–6.

[4] P. Bellavista, A. Corradi, M. Fanelli, and L. Foschini, “A survey of context data distribution

for mobile ubiquitous systems," *ACM Computing Surveys*, vol. 44, no. 4, pp. 24:1–24:45, 2012.

[5] P. Bellavista, C. Giannelli, S. Lanzone, G. Riberto, C. Stefanelli, and M. Tortonesi, "A middleware solution for wireless iot applications in sparse smart cities," *Sensors*, vol. 17, no. 11, 2017.

[6] T. Buchholz, A. Küper, and M. Schiffers, "Quality of context: What it is and why we need it," in *In Proceedings of the 10th Workshop of the OpenView University Association: (HPOVUA)*, 2003.

[7] D. Cabral Nazario, I. Vilas Boas Tromel, M. Ribeiro Dantas, and J. Leomar Todesco, "Toward assessing quality of context parameters in a ubiquitous assisted environment," in *Computers and Communication (ISCC), 2014 IEEE Symposium on*, June 2014, pp. 1–6.

[8] J.-P. Calbimonte, S. Sarni, J. Eberle, and K. Aberer, "Xgsn: An open-source semantic sensing middleware for the web of things," in *TC/SSN@ ISWC*, 2014, pp. 51–66.

[9] D. Conzon, T. Bolognesi, P. Brizzi, A. Lotito, R. Tomasi, and M. A. Spirito, "The virtus middleware: An xmpp based architecture for secure iot communications," in *2012 21st International Conference on Computer Communications and Networks (ICCCN)*, July 2012, pp. 1–6.

[10] B. de Souza and M. Endler, "Coordinating movement within swarms of uavs through mobile networks," in *IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, March 2015, pp. 154–159.

[11] 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 *Proc. of Posters and Demos Track, International Middleware Conference*. ACM, 2011, pp. 2:1–2:2.

[12] M. Endler, J.-P. Briot, V. Almeida, R. Reis, and F. Silva, "Stream-based reasoning for iot applications: Proposal of architecture and analysis of challenges," *International Journal of Semantic Computing (IJSC)*, vol. 11, no. 3, pp. 325–344, 2017.

[13] M. Endler, A. Silva, and R. Cruz, "An approach for secure edge computing in the internet of things," in *2017 1st Cyber Security in Networking Conference (CSNet)*, Oct 2017, pp. 1–8.

[14] R. Golchay, F. L. Mouël, and S. Frénot, "Towards bridging iot and cloud services: Proposing smartphones as mobile and autonomic service gateways," *arXiv preprint arXiv*, vol. 1107.4786., 2011.

[15] B. Gomes, L. Muniz, F. Silva, D. Santos, R. Lopes, L. Coutinho, F. Carvalho, and M. Endler, "A middleware with comprehensive quality of context support for the internet of things applications," *Sensors*, vol. 17, no. 12, december 2017.

[16] W. He, J. Zhang, M. Ma, and P. Wang, "Mode: A context-aware iot middleware supporting on-demand deployment for mobile devices," in *IEEE 23rd International Conference on Parallel and Distributed Systems (ICPADS)*, Dec 2017, pp. 81–88.

[17] I.O.Vasconcelos, R. Vasconcelos, B. Souza, M. R. Jr., M. Endler, and M. C. Jr, "Smartphone-based outlier detection: A complex event processing approach for driving behavior detection," *Journal of Internet Services and Applications*, vol. 8, no. 13, pp. 1–30, 2017.

[18] J. Viterbo and V. Sacramento and R .Rocha and G. Baptista and M. Malcher and M. Endler, "A middleware architecture for context-aware and location-based mobile applications," in *32nd Annual IEEE Software Engineering Workshop (SEW'08)*, 2008, pp. 52–61.

[19] Q. Jing, A. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the internet of things: perspectives and challenges," *Wireless Networks*, vol. 20, no. 8, pp. 2481–2501, Nov 2014.

[20] D. Luckham, *The Power of Events*. Addison-Wesley, 2001.

[21] A. H. Ngu, M. Gutierrez, V. Metsis, S. Nepal, and Q. Z. Sheng, "Iot middleware: A survey on issues and enabling technologies," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 1–20, 2017.

[22] G. Pardo-Castellote, "Omg data-distribution service: Architectural overview," in *Proceedings of the IEEE Conference on Military Communications - Volume I*, ser. MILCOM'03. IEEE Computer Society, 2003, pp. 242–247.

[23] L. Pitta and M. Endler, "Market design for iot data and services the emergent 21th century commodities," in *IEEE Symposium on Computers and Communications*, June 2018.

[24] 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, 2016.

[25] L. D. N. Silva, M. Endler, and M. Roriz, "MR-UDP: Yet another reliable user datagram protocol, now for mobile nodes," Depto. de Informática, PUC-Rio, Monografias em Ciencia da Computacao 06/2013, May 2013.

[26] L. D. N. Silva, M. Endler, and M. Roriz Jr., "MR-UDP: Yet another reliable userdatagram protocol, now for mobile nodes," Departamento de Informática, PUC-Rio, Tech. Rep., 2013. [Online]. Available: https://goo.gl/eQ81zs

[27] L. Silva, R. Vasconcelos, L. Alves, R. Andre, and M. Endler, "A dds-based middleware for scalable tracking, communication and collaboration of mobile nodes," *Journal of Internet Services and Applications*, vol. 4, no. 1, pp. 1–15, 2013.

[28] L. Silva, R. Vasconcelos, L. Alves, R. Andre, B. G., and M. Endler, "A large-scale communication middleware for fleet tracking and management," in *Anais do Salão de Ferramentas do SBRC*, may 2012, pp. 964–971.

[29] L. Talavera, M. Endler, and F. Silva, "Monitoring co-movement of smart objects using accelerometer data," in *Pervasive Computing and Communication Workshops (PerCom Workshops), 2015 IEEE International Conference on*, March 2015, pp. 214–216.

[30] S. Valim, M. Zeitune, B. Olivieri, and M. Endler, "Middleware support for generic actuation in the internet of mobile things," *Open Journal of Internet Of Things (OJIOT)*, vol. 4, no. 1, pp. 24–34, 2018, special Issue: Proceedings of the International Workshop on Very Large Internet of Things (VLIoT 2018) in conjunction with the VLDB 2018 Conference in Rio de Janeiro, Brazil. [Online]. Available: https://www.ronpub.com/ojiot/OJIOT_2018v4i1n03_Valim.html

[31] I. Vasconcelos, R. Vasconcelos, C. Seguin, G. Baptista, and M. Endler, "Desenvolvendo aplicações de rastreamento e comunicação móvel usando o middleware sddl," in *Proc. of the 31st Brazilian Symposium on Networks and Distributed Systems - Salão de Ferramentas*, May 2013.

[32] R. Vasconcelos, L. Silva, and M. Endler, "Towards efficient group management and communication for large-scale mobile applications," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2014 IEEE International Conference on*, 2014, pp. 551–556.

AUTHOR BIOGRAPHIES

Dr. Markus Endler is Associate Professor of Informatics at PUC-Rio and is the Principal Investigator and Head of LAC. He is also member of the steering Committee of INCT InterSCity. His research interests include mobile and pervasive distributed systems and IoT, stream processing and reasoning, and Emotive Computing.

Dr. Francisco Silva e Silva is Associated Professor at the Federal University of Maranhão, and leads the LDSi lab. His research interests include distributed and ubiquitous systems with current emphasis in IoT and Smart Cities middleware.