The Mobile Hub Concept: Enabling applications for the Internet of Mobile Things

L.E. Talavera, M. Endler, I. Vasconcelos,
R. Vasconcelos, M. Cunha
Department of Informatics
Pontifícia Universidade Católica do Rio de Janeiro
{lrios,endler,ivasconcelos,rvasconcelos,mcunha}@inf.puc-rio.br

Francisco J. da Silva e Silva
Department of Informatics
Federal University of Maranhão
fssilva@deinf.ufma.br

***Abstract*—Few studies have investigated and proposed a middleware solution for the Internet of Mobile Things (IoMT), where the smart things (Smart Objects) can be moved, or else can move autonomously, and yet remain accessible and controllable remotely from any other computer over the Internet. Examples of mobile Smart Objects include vehicles of any kind, wearable devices, sensor tags, mobile robots, etc, anyhing with embedded sensors and/or actuators. In this context of general mobility of objects, mobile personal devices (smart phones, tablets, etc.) are well suited as the universal providers of Internet connectivity and location information for simpler smart objects that lack location sensors and have only short-range wireless interfaces. This paper describes *Mobile Hub (M-Hub)*, a generic mobile middleware for IoMT, its design and prototype implementation for Android and Bluetooth. The Mobile Hub extends our previous mobile-cloud communication middleware SDDL, so that it is able to provide scalable and reliable mobile communication and data processing capabilities to mobile smart objects. Preliminary experiments have shown that our implementation of M-Hub delivers good mobility responsiveness and that the concept is suitable for IoT applications that require opportunistic discovery and connection to a variety of mobile Smart Objects.**

Keywords*—*Internet of Things, middleware, mobile objects, Mobile Hub, remote sensing and actuation.

I. Introduction

Despite the huge number of potential applications and the increasing proliferation of appliances with embedded processing and wireless communication capacity, yet there is no widely accepted approach, established standards and consolidated technologies for the Internet of Things at global scale. In other words, Internet-wide communication and processing of data from tens of billions of sensors and actuators within devices and smart objects is still a challenge. In particular, very few studies have focused on the Internet of Mobile Things (IoMT), in which the connectable things (or Objects) can be moved or can move independently, and yet remain remotely accessible and controllable from anywhere in the Internet. Mobile Objects (M-OBJs) may have very different size, purpose and complexity - they may range from 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.

The Internet of (Mobile) Things are already having a strong impact in several domains, such as smart cities and homes, environmental monitoring, health care, energy management, asset monitoring, logistics, etc. To illustrate some applications scenarios, consider the installation of wireless Air Monitoring Stations in public transport vehicles (e.g. buses, trams, etc.) equipped with CO, NO_2, SO_2, and Lead sensors. Combined with the GPS data of the vehicles, or the passenger's smart phones in these vehicles, this mobile infrastructure could provide a more accurate an timely picture of the current air pollution indices of a city than with monitoring based solely on few stationary stations. As another example, consider the delivery of goods and merchandise that require specific minimal transportation and storage conditions on their routes from producer to consumer. For example, meat and some fruits need ambient temperatures in the range of 3 to 10 degrees Celsius, or else, special flowers and plants should be in ambients with air humidity above a certain level, livestock requires smooth movement as well as places with sufficient air circulation (i.e., O_2 concentration), etc. By placing some M-OBJs with adequate sensors close to such goods, and having the sensor values probed in all the stages of transportation and intermediate storage, one could monitor the ambient and movement conditions of these and other "sensitive" merchandise, along all their transport path.

In this context of general and unrestricted mobility of Smart Objects, the main challenge is to ensure best possible connectivity with their sensors/actuators, fast discovery and connection, seamless handovers, and continuous tracking and access to the M-OBJ's resources, capability-based selection, as well as management of data streams to and from the M-OBJs in efficient and scalable way. Hence, our research aims to investigate several research challenges associated with communication and processing in the Internet of Mobile Things (IoMT), such as, how to optimize the opportunistic connectivity with M-OBjs.

IoT is evolving towards a heterogenous network including a mix of IP-based connectivity and an array of short-range, last100m wireless technologies (e.g., Bluetooth, NFC, ANT+), the latter used by peripheral devices in the Edge networks, the Objects. Moreover, according to Francis daCosta [2], IPv6 does not solve all IoT problems because management, rather than addressing and routing, are the biggest challenge of IoT. In fact, IP-based protocols will neither be supported by the vast majority of Smart Objects, nor will their over-provisioned

978-1-4799-8425-1/15/$31.00 ©2015 IEEE

and reliable services be suited to most IoT applications. The reason is that IP-based protocols *"...are intrinsically designed for high-duty cycles, large data streams, and reliability."*, while in IoT communication involves small but frequent messages, where each message individually is unimportant, but the statistical properties of the corresponding data flows carry the relevant pieces of information. Moreover, the IoT networking will not have a flat Peer-to-Peer architecture as some believe, since *"...many devices at the edge of the network have no need to be connected with other devices at the edge of the network..."*, and since *"...the communications intelligence and functionality does not exist within the end devices, other devices - propagator nodes - must be present in the network to transport data efficiently and manage the data flows..."* [2].

Considering that personal mobile devices (smart phones, phablets and tablets) and mobile Internet are becoming increasingly ubiquitous[1], more affordable and powerful, and that opportunistic and intermittent connectivity will become common place in a world filled with mobile, wearable and embedded technology (but the data streams, rather than individual data samples or messages, will be of importance), such mobile personal devices become the natural candidates to be the propagator nodes (i.e. gateways to the Internet) for the simpler IoT objects. This led us to propose the concept of *Mobile Hub (M-Hub)*, a general middleware service responsible for discovering and opportunistically connecting a myriad of simple M-OBJs (sensors/actuators) accessible only through short-range WPAN technologies to the Internet. Moreover, the M-Hub will provide information about the (approximate) location of the M-OBJs that it finds in its vicinity.

By supporting the Internet of Mobile Thing (IoMT) through this concept of M-Hub we are, in fact, coping with a more general - and harder - problem than traditionally addressed by the IoT community, which usually assumes that both the peripheral/edge devices and the propagator nodes are stationary (e.g. in smart buildings/homes or sensor networks). at some place. Essentially, the IoMT paradigm considers any situation in which the *relative position and speed* between the M-OBJs and the M-Hub is variable and may change anytime, and where M-OBJs may be reached through different and even multiple M-Hubs over time. Therefore, IoMT encompasses all situations where: (i) the M-OBJs are permanently associated with a place and the M-Hub moves across the places to opportunistically interact with the M-OBJs, e.g. for sampling local sensor data; (ii) the M-OBJs are attached to movable items, while the M-Hub is linked to a place, e.g. a warehouse, and the M-OBJs reveal their presence to the M-Hub whenever they get close to it and; (iii) one or more M-OBJs stay in co-movement with the M-Hub for a certain period of time, e.g. goods in a container, passengers in a vehicle, health-sensors carried by a patient, etc.

Of course, this more generic model of IoMT carries the burden of much more indeterminism, in the form of unpredictable sensor/actuator availability, less reliability, more connectivity volatility, higher probability of interferences, etc. Nevertheless, we believe that there exist several non-critical IoT applications which depend on the spontaneous - and intermittent - access to simple wearable gadgets or embedded sensors/actuators in mobile objects, and that may benefit from the fact that any smartphone/tablet may be used to connect them to the internet. In particular, these applications may benefit from another important characteristics of the M-Hub: its ability to enrich the M-OBJ's sensor data with contextual information obtained from the M-Hub's local sensors: its current geographic position, speed, acceleration etc. This feature will open up to applications new ways of classifying, filtering or searching data gathered from the M-OBJs. Finally, it is worth noting that in our approach the M-Hub mobility is just an option: M-OBJs and M-Hubs may also be deployed in a rigid configuration, and be associated with a specific static place, in applications such as Smart Home/Buildings, etc. Even in these cases, the M-Hub may still be attractive because it is based on a widely used and affordable technology: conventional smartphones.

The remainder of this paper is structured as follows. In the next section we give an overview of our previous communication middleware SDDL for mobile nodes, and show how the M-Hub extends it to support communication and processing for IoMT. In section III, we present the concept and main functions of the Mobile Hub, as well as its general architecture and its main components. In Section IV we describe some details our current M-Hub prototype and results of performance tests done so far. In Section V related work is discussed, and VI contains a discussion of other roles that the M-Hub may assume in future applications. Finally, in Section VII, we draw concluding remarks and point to future work.

II. SDDL Middleware

The Scalable Data Distribution Layer (SDDL) [9] is a communication middleware that connects mobile nodes (smart phones or tablets, with a wireless Internet connection) to stationary nodes in a wired core network (the SDDL Core), executing in the cloud or cluster. In our approach towards IoMT, the mobile nodes take the role of M-Hubs for connecting a variety of possible Smart Objects to the Internet, as shown in Figure 1.

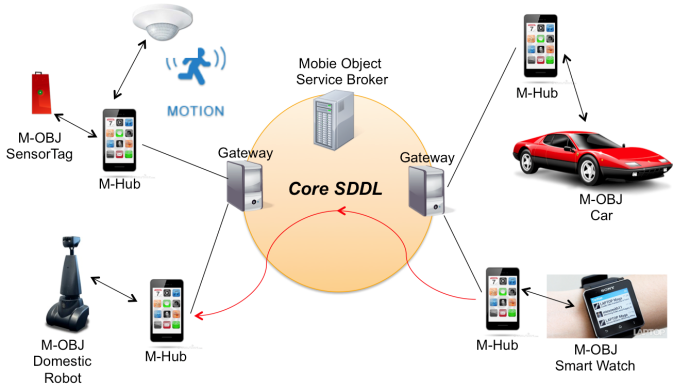

Figure 1. The extended SDDL: M-Hubs are connected both to the SDDL Core and to nearby M-OBJs, for opportunistically transmitting their sensor data and/or remote actuation commands.

The basic SDDL employs two communication protocols: the Data Distribution Service (DDS) Real Time Publish/-Subscribe Protocol for the wired communication within the SDDL Core, and the Mobile Reliable UDP (MR-UDP) for the inbound and outbound communication between the core

[1]According to eMarketer, in 2014 we already had 1.75 billion smartphone users - almost 24% of the world population!

network and the mobile nodes, including M-Hubs. DDS [7] is an OMG standard that specifies a peer-to-peer middleware architecture for real time and high-performance data distribution, with Quality of Service (QoS) contracts between producers and consumers of data (e.g., reliable communication, data persistency, priority lanes, etc.). MR-UDP, on the other hand, is Reliable-UDP with mechanisms for tolerating intermittent connectivity, dynamic IP address changes of the Mobile nodes and reaching these nodes behind firewalls/NATs. It is used by the mobile nodes to connect with a special type of SDDL Core node called *Gateway (GW)*, of which any number can be deployed in the SDDL Core. Each Gateway maintains one independent MR-UDP connection with each mobile node, and is responsible for translating application-messages from MR-UDP to the intra-SDDL core protocol, and, in the opposite direction, converting SDDL Core messages to MR-UDP messages and delivering them reliably to the corresponding mobile nodes. Any mobile node uses the *ClientLib*, a library used to establish and manage a MR-UDP connection with one or more Gateways and that hides most MR-UDP details and message retransmission issues from the application layer, and also supports a fully application-transparent handover of the mobile node between SDDL Gateways. The M-Hub is thus nothing else but a special kind of mobile node that opportunistically connects to M-OBJs, and supports one or more WPAN technologies: Bluetooth, BLE, NFC, ANT+, etc.

The SDDL Core includes several other specialized services in charge of load balancing, data persistency, data stream processing and group-cast communication, whose explanation can be found in papers [9], [10]. The interested reader can download a VM with pre-installed SDDL, as well as find examples and tutorials for implementing SDDL-based applications in Java, Android and Lua[2].

III. The Mobile Hub concept and its design

As already mentioned, the Mobile Hub (M-Hub) is a general-purpose middleware service (executed on a conventional personal mobile device) that discovers, registers and enables remote unicast, broadcast and group-cast mode communication to/among many kinds of Mobile Objects (M-OBJ), through the SDDL middleware. The M-Hubs thus "bridges the gap" between the Internet connection with the SDDL Core, and the short-range wireless connections established with M-OBJs. The latter may be very simple wearable devices or gadgets/vehicles/robots with embedded sensors or actuators, and with no significant processing and storage capacity. For managing uniformly the discovery and connection with nearby M-OBJs using different short-range wireless technologies, we designed the *Short-range Sensing, Presence & Actuation (S2PA)* API, a generic and technology-independent protocol which runs on the M-Hub. SP2A thus has a common interface for the different low range wireless technologies (WPAN) within the M-Hub.

A. Short-Range Sensor, Presence and Actuation API

The *S2PA* 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 technologies (WPAN). To this end, it defines some basic methods and interfaces that all these technologies should implement: 1) Discovery of, and connection with M-OBJs, 2) Discovery of services provided by each M-OBJ, 3) Read and write of service attributes (e.g., sensor values, and actuator commands) and 4) 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.

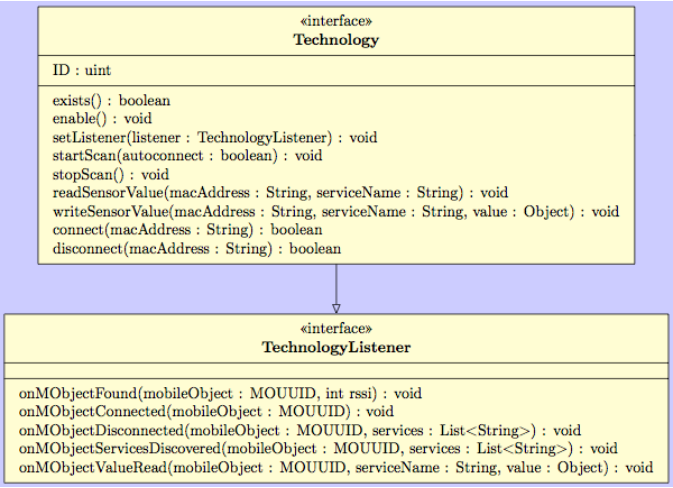

Figure 2. Main two interfaces of the S2PA

As its first realization we have implemented S2PA for Bluetooth 4.0 (a.k.a. *Bluetooth Smart/Low Energy - BLE*) and for Classic Bluetooth[3]. In fact, BLE is emerging as a very promising technology, because it is power efficient, enables fast discovery of peripheral devices and supports approx. 2500 simultaneous connections. But the most important reason for choosing BLE is the fact that it is now made available on a growing number of Android, iOS, Blackberry smart phone models. Moreover BLE is being embedded into a growing array of peripheral devices, gadgets, beacons, small Sensor Tags, etc.[4]. Despite its higher pairing/connection time, Classic Bluetooth (2.0 and 3.0) is also very important since it is supported by the majority of current peripheral devices (e.g. health and fitness devices, such as the Zephyr BioHarness 3[5]). Moreover, Classic Bluetooth also allows M-Hubs to find and communicate among them, and thus to implement different

[2]http://www.lac-rio.com/dokuwiki/doku.php?id=tutorial

[3]Classic Bluetooth was implemented because of the wide range 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 for handing over discovered nearby M-OBJs (see *Handover Manager*).

[4]See: www.bluetooth.com/Pages/Bluetooth-Smart-Devices-List.aspx

[5]http://zephyranywhere.com/products/bioharness-3/

protocols for managing handovers of M-OBjs between the available M-Hubs.

B. M-Hub's main Components

The M-Hub is multi-threaded and consists of the following four local services and two managers, all executing in background and in parallel with user apps. 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 GW 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 sample's the device's battery level and checks if it is connected to a power source. Finally, the **Handover Manager**, continuously checks some connectivity QoS parameter of each nearby M-OBJ gathered by the S2PA Service, such as the sensed signal strength WPAN RF, and according to the situation, interacts with other nearby M-Hubs, so as to proactively share information and parameters about M-OBJs, and eventually swap responsibility for handing-out or handling-in M-OBJs with these M-Hubs.

Figure 3 shows the M-Hub architecture and some details of the interactions between the aforementioned components. *SensorTag, DevType2*, etc.. are modules that handle the information received from/ sent to specific M-OBJs. Each of them possess a **convert()** method which handles the transformation of the raw data (bytes array) received from the M-OBJ's sensors to an array of doubles. The aforementioned conversion is specific for each type of M-OBJ.

The *S2PA Service* is capable of managing several WPAN technologies by calling generic methods that are mapped to their corresponding WPAN-specific classes. As soon as a new M-OBJ is detected, it starts getting data from it, which are then handed over to the *Connection Service*, where they are stored in a *msg buffer*. The ConnectionService may also interact

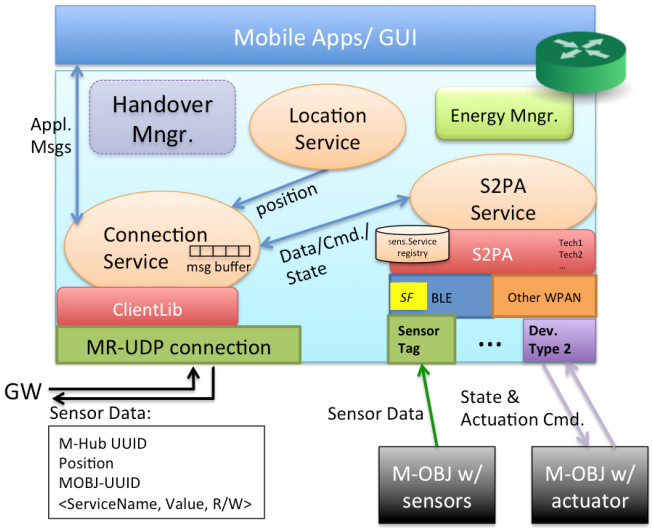

Figure 3. M-Hub's main components while it interacts with two M-OBJs, with different WPAN technologies.

with any mobile app executing on the smartphone (the upper tier), forwarding application messages between the SDDL Core and the apps, in both directions (see left-side vertical arrow). The *Location Service* gets the latest geo-coordinate of the M-Hub, which is attached to any outbound data by the *Connection Service*. In order to save Internet bandwidth and energy, in some cases, the M-Hub can perform some basic local processing (summarization, filtering) over the stream of sensor data gathered from the nearby M-OBJs and from the M-Hub's local sensors. Or else, it may continuously compare accelerator data and check if some M-OBJs (w/ accelerometer sensors) are in co-movement with the M-Hub. Finally, the *Energy Manager* sets the frequency (of discovery, update and transmission) cycles for all the three Services, in accordance to the current battery level, and if the device is plugged or not to the power source.

IV. Current Status and Preliminary Results

Our current M-Hub prototype already implements the first six mandatory functionalities (c.f., Section III) and has *S2PA* for Bluetooth LE and Classic. It discovers, connects and starts receiving sensor data updates from any number of nearby M-OBJs. We tested it with off-the-shelf SensorTags[6]. Figure 4 shows the M-Hub viewer app displaying four connected SensorTags (red icons).

A. Preliminary Performance Tests

We already did some preliminary experiments to access the latency of the discovery and connection activities of the M-Hub using BLE with four Sensor Tags, and obtained encouraging results. For these experiments, we configured S2PA to perform a WPAN scan every 3 seconds (energy level HIGH), and set the scan duration at 2 seconds. We used a Motorola Moto X smartphone running Android 4.4.2 KitKat. The notebook used to run the SDDL Core (Gateway and WebMonitor) for all the

[6]Texas Instruments CC2541 Sensor Tag - http://www.ti.com/lit/ml/swru324b/swru324b.pdf

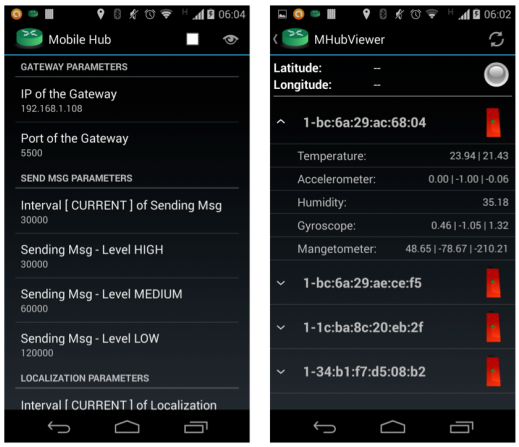

Figure 4. Screenshot of the M-Hub for Android with 4 connected SensorTags.

experiments was an ASUS Intel(R) Core(TM) i7-4500U CPU 1.80GHz with 5857 MB of RAM, running Arch Linux (Kernel 3.16.3-1-ARCH). The type of WLAN used is IEEE 802.11bgn.

We measured the Connection Time (CoT), Services Discovered Time (SDT), and Enable Notification Time (ENT), in seconds, both for the first connection of the M-Hub with the M-OBJs, and for the follow-up reconnections (Table I). We ran each experiment 12 times and calculated the mean value and the standard deviation. It is also important to remark that the behavior of BLE in Android is synchronous, so the operations like connections to each of the M-OBJs is done sequentially.

First Connection	CoT (s)	SDT (s)	ENT (s)
Mean value	0.21192	9.132	0.325
Std. Deviation	0.1578	0.14381	0.15839
Reconnections	**CoT(s)**	**SDT (s)**	**ENT (s)**
Mean value	0.2325	0.15517	0.34625
Std. Deviation	0.10007	0.01609	0.0953

Table I. PERFORMANCE OF M-OBJ DISCOVERY AND CONNECTION TIMES FOR BLUETOOTH LE

Services discovery in BLE is the slowest operation. And since it will find all the services, characteristics and descriptors that a M-Object possesses, its delay will be proportional to the number of services, which in the case of our SensorTags was 6 sensors. However, we found that this Services Discovery time decreases very much at later reconnections, as can be seen from Table I.

Further experiments were realized to measured the amount of time that it takes for a node in the the SDDL Core to receive the sensor data from one M-Hub (MHUB2) that established a connection with a M-Object (i.e., until services are discovered) since another M-Hub (MHUB1) was disconnected from the M-OBJ. This time interval was measured at the server, from the moment it receives *M-OBJ DisconnectMessage* from MHUB1 until it receives message that the same M-OBJ services have been (re)-discovered. The Motorola Moto X was used as the MHUB1 and a Moto E with Android 4.4.4 KitKat was used as MHUB2, and both communicated through IEEE 802.11bgn with the notebook executing the server in the SDDL Core. Unlike the former experiments, in these tests the interval between consecutive WPAN scans was set to zero seconds. Consistent to the previous results, the connection time of MHUB2 with the M-OBJ has averaged 10.32442 seconds for a first connection, and averaged 0.92517 seconds for follow-up reconnections, with a standard deviation of 1.00115 and 0.95336 seconds respectively.

The experiments show that M-Hub's BLE-specific implementation of S2PA enables a very low, almost instant *connection+service discovery* latency of M-OBJs (SensorTags), specially if the M-Hub had already been connected to the M-OBJs before. Hence, for any IoMT application that requires sensor data of M-OBJs to be collected and transmitted to a node in SDDL core in real time **by any M-Hub that happens to be in its BLE wireless range**, then it will suffice for a M-Hub to stay only 10 seconds within the BLE coverage of the M-OBJ. In this case, The M-Hub will already be able to receive and transmit the M-OBJ's sensor data to the server. Considering that BLE's communication range is approx. 20-30 meters, this means that if the relative speed between the M-Hub and the M-OBJ is ≤ 3 m/s, then there is high probability that the M-Hub will manage to transmit some M-OBJ sensor data to the server. And the chances will even be higher if there are more than one M-Hub in the vicinity of the M-OBJ.

V. RELATED WORK

Most of the works, in both industry and academia, on IoT in mobile settings are based on fixed communication infrastructures and utilize a client-server model, where mobile clients issue range queries to central query processing servers or brokers using protocols like MQTT over WiFi or ZigBee. In these systems, the stationary objects continuously transfer their data to a central server and clients can obtain information about near-by objects by issuing queries to the central server [5].

Some of the recent works address peer-to-peer and mesh network connectivity with service discovery in a mobile environment. There are industry protocols like Thread, from TheadGroup, AllJoin that are specialized on support of discovery and interoperability of things across devices and platforms in specific environments (e.g. home, office), and academia protocols designed to be mobile and energy efficient by grouping devices in a neighborhood such that devices in a group will take turns to announce the existence of other devices in a group [4]. There are also technologies like Bluetooth Service Discovery Protocol (BSDP) Universal Plug and Play (UPnP) [6] and Secure Service Discovery Service (SSDS) [1] as a specification which enables close devises to communicate with each other at low cost and low power consumption, defines interactions among smart object, people, and environments and also supports authentication, privacy, and integrity.

Several efforts also have been based on the CoAP protocol, which natively provides a mechanism for service location-based discovery [8]. Each CoAP server must expose an interface to which the generic node can send requests for discovering available resources. The CoAP server will reply with the list of resources and, for each resource, with an attribute that specifies the format of the data associated to that

resource. CoAP, however, does not specify how a node joining the network for the first time or how it must behave in order to announce itself to the resource directory node.

Although there are many approaches for IoT, many of them don't embrace mobile nodes, do not consider movable smart objects, or do not scale. We are unaware of a comprehensive approach focused on the Internet of *Mobile Things*, in which the connectable things can be moved or can move independently, and yet remain remotely accessible and controllable from anywhere in the Internet with handover. Nor have we seen any work showing the design, concrete prototype implementation and experiments of a Mobile Hub. with a concrete WPAN technology. The only work that presents a similar idea of using the smartphone as an IoT Gateway is [3], but their software architecture for the smartphone is rather high-level, uses traditional protocols (e.g., TCP, UDP), and does not consider any concrete short-range, low-power WPAN or WLAN technology.

VI. Discussion

For IoMT applications, most relevant information about M-OBJs are its current location and its current state of movement. However, most of these objects (simple peripheral devices) don't have this information about their position/movement, and also are not connected to the Internet. Thus, there is the problem of how remote clients can learn the M-OBJs whereabouts, track their movements, or else, interact directly with them for sensing or actuation purposes. To use the *Mobile Hub* concept for this purpose is surely only a partial solution to this problem, because the proximity - and reachability - of a M-OBJ through a M-Hub cannot be guaranteed in general, but is always dependent on occasional discovery and opportunistic interactions between them. Nevertheless, the M-Hub concept remains a viable - and very economic - solution whenever smart phones can be set up as dedicated propagators forsome M-OBjs.

Although the general IoMT connectivity problem is still open, one of our goals in enabling IoMT through the M-Hub concept has been to maximize the chances that a M-Hub can connect the M-OBJs, given that they stay mutual WPAN coverage for some minimal period of time (e.g. a few seconds). And such optimization necessarily requires efficient S2PA implementations, prioritized message handling in the M-Hub (M-OBJ connection/disconnection detection), effective inter M-Hub handover protocols and support for WPAN technologies with reasonable communication range, efficient discovery and low connection times. Regarding these requirements, we think that our current prototype is already a good starting point towards making the connection with M-OBJs as agile as possible. Moreover, we are aware that much of the credit for the good performance results of our current prototype goes to Bluetooth LE, which in our view is one of the most promising WPAN technologies IoMT due to its well-known energy and connectivity efficiency.

VII. Conclusion

The *Mobile Hub* concept is independent of the mobile platform and is designed to be extensible to support different WPAN technologies since its constituent *S2PA Service* implements a generic service with a uniform interface that can be mapped to each supported WPAN technology. Our first prototype was implemented for Android, uses our mobile communication middleware SDDL, and has used Bluetooth Low Energy (BLE) and Classic Bluetooth as the showcase WPANs. In fact, BLE turned out to be an excellent choice as WPAN for IoMT due to its connectivity and energy efficiency, its increasing adoption for smart things/objects, and because is being supported by most smartphone brands.

In preliminary experiments the M-Hub prototype exhibited excellent results for discovery, reconnection and handover of BLE-enabled M-OBJs. For example, if a M-Hub and a M-OBJ have already discovered each other before, the subsequent reconnection takes 0.9 seconds, and can be done for 4 M-OBJs happens in less than 5 s. In spite of encouraging initial results, we are aware that our M-Hub prototype is only "scratching the surface of IoMT", and much interesting research, software development and applications can be derived from this work. In particular, our future work will investigate the problems and possible approaches for inter-M-Hub handover protocols, implementation of local *Data (Stream) Processing* capabilities into the M-Hub, so that it is capable of processing the sensor data received from nearby M-OBJs, deriving and transmitting only some higher-level information/events from this sensed data to the cloud, and the possible implementation of ANT+ as a new WPAN, since it is becoming available in several Android devices.

References

[1] S. Czerwinski, B. Y. Zhao, T. Hodes, A. Joseph, , and R. Katz. An architecture for a secure service discovery service. In *Fifth Annual International Conference on Mobile Computing and Networks (MobiCom '99)*, 1999.

[2] F. DaCosta. *Rethinking the Internet of Things: a scalable approach to connecting everything*. ApressOpen, New York, NY, 2013.

[3] 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*, 1107.4786., 2011.

[4] P. Huang, E. Qi, M. Park, and A. Stephens. Energy efficient and scalable device-to-device discovery protocol with fast discovery. In *IEEE International Conference on Sensing, Communications and Networking (SECON 2013)*, pages 1–9, 2013.

[5] U. Hunkeler, H. Truong, and A. Stanford-Clark. Mqtt-s - a publish/-subscribe protocol for wireless sensor networks. In *3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE)*, pages 791–798, 2008.

[6] B. A. Miller, T. Nixon, C. Tai, and M. D. Wood. Home networking with universal plug and play. *IEEE Communications Magazine*, pages 104–109, 2001.

[7] G. Pardo-Castellote. Omg data-distribution service: Architectural overview. In *Proceedings of the 2003 IEEE Conference on Military Communications - Volume I*, MILCOM'03, pages 242–247, Washington, DC, USA, 2003. IEEE Computer Society.

[8] Z. Shelby, K. Hartke, , and C. Bormann. Constrained application protocol (coap), 2013.

[9] 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*, 4(1):1–15, 2013.

[10] R. O. 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*, pages 551–556, March 2014.