A Comprehensive and Scalable Middleware for Ambient Assisted Living Based on Cloud Computing and IoT†

Berto de Tácio Pereira Gomes 1 2*, Luiz Carlos Melo Muniz 1; Francisco José da Silva e Silva 1; Luis Eduardo Talavera Ríos 3 and Markus Endler 3

1 Universidade Federal do Maranhão (UFMA), Brazil
2 Instituto Federal do Maranhão (IFMA), Brazil
3 Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), Brazil

SUMMARY

Ambient Assisted Living (AAL) main goal is the development of health monitoring systems for patients with chronic diseases and elderly people through the use of body, home and environmental sensors that increase their degree of independence and mobility. A comprehensive software infrastructure for AAL systems should be able to cover scenarios involving several patient mobility levels, locations, and physical and cognitive abilities. Cloud computing can provide to AAL systems the ability to extend the limited processing power of mobile devices, but its main role is to integrate all stakeholders through the storage and processing of health data and the orchestration of healthcare business logic. On the other hand, the Internet of Things (IoT) provides the ability to connect sensors and actuators, integrating and making them available through the Internet. This paper presents the Mobile-Hub/SDDL, a middleware for AAL based on cloud computing and IoT. We discuss how this middleware can handle the requirements of the main health monitoring scenarios and present results that demonstrate the ability to opportunistically discover and connect with sensors in a timely manner and the scalability necessary for handling the connection and data processing of many connected patients.

Copyright © 2016 John Wiley & Sons, Ltd.

1. INTRODUCTION

In healthcare, Ambient Assisted Living (AAL) has been used to designate a multidisciplinary research field, whose main efforts focus on developing intelligent systems for monitoring the daily activities of patients residing or transiting in smart environments, such as smart homes [1]. The most common patients using AAL systems are those who have some type of chronic disease such as arrhythmia, sleep apnea, diabetes or chronic obstructive pulmonary disease. AAL systems allow physicians, caregivers and family members to remotely monitor a patient health status, providing her/him more autonomy and mobility during treatment. In this way, AAL systems promote a transition from the traditional model of healthcare centered on organizations (e.g. hospital) to a patient-centered model, in which predominate individual monitoring services at home or even in

*Correspondence to: Universidade Federal do Maranhão, Programa de Pós-Graduação em Engenharia de Eletricidade (PPGEE), Centro de Ciências Exatas e Tecnologia (CCET), Avenida dos Portugueses s/n, Campus Universitário do Bacanga, CEP: 65085-580, São Luís, Maranhão, Brazil, phone +55(98) 98819435, email: bertodetacio@lsdi.ufma.br
†Please ensure that you use the most up to date class file, available from the CPE Home Page at http://www3.interscience.wiley.com/journal/117946197/grouphome/home.html
mobile scenarios, such as during a walk in a city park or a trip. There is a wide range of other applications for AAL systems, such as rescue and emergency response systems, fall detection, video surveillance systems, etc. Nowadays, AAL systems are regarded as a trend in a context of increasing awareness of how the Internet can be used to personal healthcare.

AAL systems are composed of several technologies: sensors and actuators, portable/wearable devices, heterogeneous wireless networks, medical applications executing on mobile devices (handhelds), personal computers, or in a cloud computing infrastructure. Among the variety of low-level sensors that can be applied in AAL systems, there are the wearable medical sensors, able to collect data from physiological signals (e.g. ECG, EMG, heart rate, oxygen consumption) or data reflecting the body movement (e.g. accelerometer). Personal mobile devices, such as smartphones, are also usually equipped with motion and location sensors (e.g. accelerometer, GPS). Environmental sensors can also be used, as they collect information that helps determine if environmental conditions (e.g. temperature, light, humidity, carbon dioxide levels) favor or not the patient’s health. In addition to gathering data from sensors, AAL systems may also be used to control medical devices used by patients, such as insulin pumps. A software infrastructure for such health monitoring systems will typically run on a computer connected to the Internet, either a mobile personal device (e.g., tablet, smartphone) or a personal computer (e.g. desktop, laptop), so that the sensor data can be transmitted to medical software executing in the cloud (e.g. for analysis, recording and producing alerts to healthcare professionals and caregivers). Thus, these patients’ computers on one hand communicate with the body and environmental sensors by means of some short-range wireless technologies (e.g. Bluetooth, Zigbee, ANT+), and on the other hand act as residential gateways that enable communication from sensors to remote healthcare applications via some wireless telecommunication technologies (e.g., GPRS, 3G/4G, WLAN) [2].

AAL is an emerging research field for which there are still many challenges related to patient monitoring, such as:

1. **Comprehensiveness of scenarios**: this challenge is related to the difficulty of developing an AAL system able to comprehend the whole range of scenarios where health monitoring can be applied. For example, Varshney [3] describes several AAL scenarios: Home Care, Mobile Health, Nursing Home and Assisted Living Facilities. These scenarios differ in aspects related to the patient, such as levels of user mobility; degrees of physical and cognitive abilities; user location; and distance to healthcare professionals. The point is that a software infrastructure for AAL is comprehensive if it is capable of supporting several AAL scenarios, allowing patients to move from one scenario to another without any disruption of the available AAL services and without data loss. In addition, each AAL scenario has specific challenges, addressed in details in Section 4.

2. **Reliable Communication**: for Varshney [4], one of the most difficult challenges in AAL systems using infrastructure-oriented wireless networks (WLANs, satellites, and cellular networks), especially in emergency cases, is the reliability of message delivery. The unpredictable quality and reliability could lead to difficulty in achieving continuous health monitoring and delivery of monitoring signals and other information from a patient to healthcare professionals. Eventually, the delayed medical response to patients could result in danger to them. To support the reliability and monitoring delay requirements of health monitoring, a significant effort is necessary in creating protocols to support routing and reliable delivery of messages carrying patient information.

3. **Heterogeneous Technologies**: the challenge is to achieve a flexible software infrastructure able to interact with different types of sensors and actuators that may be accessed through different short-range communication technologies. Moreover, data packets from different sensors may have distinct formats and encodings, raising the necessity of sensor transcoding modules that encapsulate the logic necessary to interpret the exchanged data. However, since it is unfeasible to have built-in transcoding modules for all possible sensor devices that a patient may interact with as he/she moves around different environments, it is necessary to have some sort of dynamic/on-demand deployment of code.
4. **Scalability:** this challenge is related to the current great demand for AAL services. AAL systems must be prepared to support a large number of users. This implies in having to deal with a large amount of connections from the users’ mobile devices to the AAL servers and also the capacity for handling a large volume of requests aiming the storage or processing of the patient data collected through sensors. Scalability, in this context, should be understood as the ability of the AAL system to continue meeting its requirements even in face of an increased demand for its services.

5. **Power Management:** Since at least part of an AAL system usually runs on mobile devices that have a limited source of energy due to the use of batteries, it is important to carefully handle the energy consumption of the provided AAL services. The provision of adaptive behavior in response to energy availability is usually required to deal with the power management challenge.

The integration of the Internet of Things (IoT) and cloud computing into AAL systems can provide several benefits and help to deal with some of above challenges. We next present some concepts of these paradigms and discuss why this integration is important for the development of patient monitoring systems.

IoT is an emerging paradigm that integrates approaches and technologies from areas such as ubiquitous and pervasive computing, Internet protocols, sensing technologies, communication technologies, and embedded devices [5]. This integration aims to generate a global dynamic system, in which thousands of heterogeneous addressable devices, known as smart objects (S-OBJ’S), are interconnected by communication networks and are able to exchange data with each other without the need of human intervention in most of the occasions. In this view, the IoT is an approach that merges the physical and virtual worlds into a single network: the network of things. An IoT extension called IoMT (Internet of Mobile Things) proposes scenarios where the smart objects can be moved, or else can move autonomously, and yet remain accessible and controllable remotely from any other computer over the Internet [6]. Examples of mobile smart objects include wearable devices, sensor tags, mobile robots, vehicles of many kinds, anything with embedded sensors and actuators. In this context of general mobility of objects, mobile personal devices 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.

IoT is particularly useful to AAL systems because it allows patient monitoring to happen in innovative scenarios, such as the monitoring of elderly people living in smart environments (e.g. Smart Homes and Smart Cities), where sensors, actuators, and networks operate in an integrated manner in order to provide some assistance service to the monitored patients, without their explicit awareness. Advances in mobile gateways development for the IoT, which are intended to allow opportunistic interaction with sensors and actuators in mobility situations, further increase the scope and flexibility of monitoring and patient care by enabling the systems to conveniently use the services provided by smart objects that are casually discovered as the patients transit through several smart environments. In many monitoring scenarios, it is also necessary that the data collected through sensors to be sent periodically to healthcare professionals and caregivers through the Internet. In this regard, there have been several proposals for standards and communication protocols for data distribution in IoT, especially for telemetry [7].

Many features of the cloud computing model are useful for the development of remote patient monitoring systems. For example, the efficient management of large volumes of data to be produced by hundreds of thousands of sensors is an important issue for the adoption of AAL systems in scenarios involving a wide range of patients using mobile devices. In this context, AAL takes advantage of the cloud ability to scale its resources according to the patient’s demand and volume of data resulting from the monitoring. Another issue is that since mobile devices have limited memory, power, processing, and communication capacity, they require a scalable computing infrastructure with high processing performance and capacity of store bulk data that enables, both on-line and off-line, analysis of patient data [8]. An important issue for AAL systems is availability because the interruption in patient monitoring can lead to serious consequences, especially in emergency situations. The integration of the AAL application with the cloud can help meet this requirement,
because if a cloud server stops working, for example, others who are part of the infrastructure continue to provide the service. Finally, information sharing and collaborative work between health professionals and caregivers become easier because all the patient’s data are available through the cloud. Cloud services can also orchestrate the interaction between all stakeholders of the AAL system.

The contribution of this paper is to present a flexible, reliable, and scalable software infrastructure that addresses the above-described challenges of AAL systems. Moreover, the proposed software infrastructure is comprehensive in the sense that it can be applied in several AAL scenarios. It is composed of two main components: Mobile Hub (M-Hub) and Scalable Data Distribution Layer (SDDL). M-Hub is a general-purpose middleware service for IoTM that runs on personal mobile devices and is responsible for discovering and opportunistically connecting to several simple smart objects accessible only through short-range WPAN technologies, making them available through the Internet. To extend the capabilities of the mobile devices, the M-Hub uses the SDDL, a middleware for communication, processing and storage of data in clouds.

This paper is structured as follows. In Section 2 we present the main components, protocols, and services provided by the SDDL middleware, while in Section 3 we present the M-Hub. In Section 4, we present the health monitoring scenarios in which this software infrastructure can be applied and also show a case study involving the implementation of an AAL system using the M-Hub and SDDL. In Section 5, we present a detailed evaluation of the proposed software infrastructure considering two different aspects: I) discovery and connection with smart objects, and II) support for client scalability using AAL applications. In Section 6, we analyze some related work and perform a comparison between our software infrastructure with other AAL infrastructures that are also focused on health monitoring. Still in this Section, we discuss the differences between this work and our previous papers. Finally, in Section 7 we drive the conclusions of this work and highlight some future initiatives that we also intend to address.

2. THE SDDL MIDDLEWARE

The Scalable Data Distribution Layer (SDDL) is a communication middleware that connects stationary nodes in a wired core network (the SDDL cloud) to mobile nodes with an IP-based wireless/mobile data connection. Some of the stationary nodes are information and context data processing nodes, others are gateways to communication with the mobile nodes, and yet others are monitoring and control nodes operated by humans, and capable of displaying the mobile nodes’ context information (such as the location and vital signals in the case of a patient monitoring), managing groups of nodes, and sending messages to individual or groups of mobile nodes. In our approach towards IoMT, the M-Hub is a special kind of mobile node that opportunistically connects to nearby smart objects, and supports one or more WPAN technologies, such as Bluetooth, BLE, NFC, ANT+, as shown in Figure 1.

The SDDL employs two communication protocols: (i) Mobile Reliable UDP protocol (MR-UDP) for inbound and outbound communication between the core network and the mobile nodes; and (ii) DDS Real-Time Publish-Subscribe RTPS Wire Protocol for scalable communication within the SDDL core network. The nodes in the SDDL core rely on the DDS Data Centric Model, where DDS Topics are defined for communication and coordination between these core nodes. These two protocols will be further detailed in Sections 2.1 and 2.2 respectively. As part of the core network, there are four types of SDDL nodes with distinguished roles:

1. Each Gateway defines a unique Point of Attachment (PoA) for connection of the mobile nodes with the SDDL cloud. The gateway is responsible for managing a separate MR-UDP connection with each of these nodes, forwarding any application-specific message or context.
information into the core network and, in the opposite direction, converting DDS messages to MR-UDP messages and delivering them reliably to the corresponding mobile node(s). To handle concurrent connection requests and also manage the receipt of messages sent by multiple mobile clients simultaneously, each gateway uses an MR-UDP server socket and a thread pool. The server socket is used to continuously receive and accept connection requests from mobile clients. For each accepted connection request, a new socket is created. The socket is an abstraction that represents a separate connection for each mobile client. A new thread is instantiated for each connection, which will be responsible for handling the messages sent by its respective mobile client. To achieve scalability regarding the number of mobile clients connected to the SDDL cloud, new gateways can be dynamically instantiated. Thus, a greater number of connections can be accepted and handled separately. A load balance algorithm dynamically redistributes the mobile clients between the available gateways using a seamless handover procedure (without data loss). Being the handler of connections with the mobile nodes, the gateway is also responsible for notifying other SDDL core nodes when a new mobile node becomes available or when mobile nodes disconnect from it. This information is necessary for some other functionality of the SDDL core, such as the buffering of messages addressed to temporarily offline mobile nodes, for later delivery.

2. The PoA-Manager is responsible for two tasks: to periodically distribute a list of Points of Attachments (PoA-List) to the mobile nodes and to eventually request that some mobile nodes switch to a new gateway/PoA. The PoA-List is always a subset of all available gateways in SDDL, and the order in the list is relevant, i.e., the first element points to the preferred gateway/PoA and so forth. By having an updated PoA-List, a mobile node may always switch its gateway if it detects a weak connection or a disconnection with the current gateway.
Moreover, by distributing different PoA-Lists to different groups of mobile nodes, the PoA-Manager is able to balance the load among the gateways as well as announce to the mobile nodes when a new gateway is added or an existing gateway is removed (or have failed). A protocol for seamless handover of mobile nodes between gateways is implemented at the Temporary Disconnection (MTD) Service. This service is responsible for listening to disconnected-node messages produced by the gateways and, thereafter, to collect all messages that could not be delivered to the mobile node during its handover or off-line period. However, as soon as the node is connected to a new gateway, which will also be announced by the corresponding gateway, the MTD Service will resend all the buffered messages through the DDS domain to the new gateway, which will deliver them to the mobile node. Because not all applications require such reliable delivery, the MTD Service is an optional feature in SDDL.

3. **Processing Nodes** are server nodes that extend the processing power of mobile nodes, being able to perform computationally intensive tasks. Each processing node can handle requests sent by one or more mobile clients. By default, requests are executed in the order of their arrival at the processing node. However, SDDL developers can use threads to perform tasks concurrently in the same processing node. Additionally, there can be several instantiated processing nodes. In this case, an assignment function is used to determine which of them will perform a given task. Thus, a great number of requests can be executed concurrently, either by using more than one processing node and/or by exploring concurrency with each one of them, which tends to decrease the response time of the mobile client’s requests.

4. **Load Balancer** The Load Balancer is responsible for monitoring the load of processing nodes and redistributing the system workload when a workload unbalance situation is detected. For balancing this workload of data sent by the mobile nodes to the processing nodes in the core network, SDDL uses a solution suitable for DDS-based systems called Data Processing Slice Load Balancing (DPSLB) [11]. The key concept of the solution is the Data Processing Slice (DPS) concept, the basic unit of data flow allocation. A DPS is a selection of (inbound) data items where one of its attributes satisfies a given filtering condition. The general idea is that each processing node has some DPS assigned to it and that load balancing is equivalent to a redistribution of the total number of DPS among the processing nodes according to their current load (which is indicated by several metrics, such as CPU and memory utilization).

### 2.1. Mobile Reliable UDP (MR-UDP)

The Mobile Reliable UDP (MR-UDP) is the communication protocol used for the interaction between the gateway and the mobile nodes. This protocol implements TCP-like functionality at the top of UDP and has been customized to handle intermittent connectivity, Firewall/NAT traversal and changes of IP addresses and network interfaces. Each message, in either direction, requires an acknowledgment that, if not received, causes the transmission to be retried several times before the connection is considered broken. In addition, MR-UDP implements the following optimizations: a reduced number of connection-check packets; the transparent continuation of an MR-UDP connection regardless of IP address changes; a small number of connection maintenance packets for Firewall/NAT traversal; and simple data-flow control. Because the mobile device has its own restrictions, such as limited battery life, it is also important that the communication protocol does not use too much processor resources. These optimizations are very important because wireless WAN networks (e.g. 3G/4G) are not fully reliable everywhere, and resources must be used wisely and only when truly necessary. For example, when a mobile node connected to a gateway enters an area with no or weak connectivity, it may suffer a temporal disconnection; and when the connection is re-established, the mobile device will probably obtain a new IP address. Using MR-UDP, the previous connection to the mobile node will be maintained, and all buffered UDP packets will be delivered in the original order if the disconnection time is shorter than a threshold timeout, which is an MR-UDP parameter (i.e. 30 seconds).
2.2. Data Distribution Service (DDS)

DDS is an Object Management Group (OMG) standard which specifies a publish/subscribe communication infrastructure aimed at high performance and real-time distribution of critical information in distributed systems. This specification was designed with the intention of obtaining a high scalability, portability and interoperability [12]. The DDS was conceived around a Data-Centric Publish-Subscribe (DCPS) model based on topics, which makes the complex programming of distributed communication protocols transparent to the developer. Topics are typed structures that connect Publishers to Subscribers and is where it is located the information that will be exchanged on the network. Publishers and Subscribers of a DDS Domain (the collection of nodes pertaining to a single application) are containers for Data Writers and Data Readers, respectively, which exchange typed data through a common Topic. Specifically, the DCPS automatically manages the delivery of all DDS messages in a totally decoupled and asynchronous way, i.e. without requiring the application to explicitly determine which will be the sender and receiver of each message or handle message acknowledgments and retransmissions. The DCPS also supports a large array of Quality of Service (QoS) policies for communication (e.g. best effort, reliable, ownership, several levels of data persistence, data-flow prioritization) [12] [13]. Taking advantage of DDS’ distributed P2P architecture and its highly optimized Real-Time Publish-Subscribe wired protocol, SDDL is naturally scalable, i.e. new processing nodes or gateways can be dynamically added to SDDL’s core whenever more mobile nodes have to be served, or new data flow processing is required.

3. THE M-HUB

Personal mobile devices and mobile Internet are becoming increasingly ubiquitous, more affordable and powerful, and that opportunistic and intermittent connectivity will become commonplace in a world filled with mobile, wearable and embedded technology. In this way, personal mobile devices become the natural candidates for serving as propagators of IoT objects, allowing them to have an Internet representation. This led us to propose the concept of Mobile Hub (M-Hub) [16], a general-purpose middleware that extends the SDDL, responsible for discovering and opportunistically connecting a myriad of simple M-OBJs accessible only through short-range WPAN technologies to the Internet. M-Hubs “bridge the gap” between the Internet connection to the SDDL Core (e.g. executing in a cloud) and the short-range wireless connections with M-OBJs. Hence, it must have full-fledged processing power, sufficient memory, and network interfaces both to Mobile Internet (2G/3G/4G) and to low-range, low-power WPAN communication technologies. Moreover, the M-Hub add to the data provided by M-OBJs meta-data such as the current local time or/and the (approximate) location of the M-OBJs that it finds in its vicinity, which is obtained either by GPS, network or manually configured (when used in stationery settings). This feature opens up to applications new ways of classifying, filtering or searching data gathered from the M-OBJs.

In the proposed architecture, the M-Hub provides the following functionalities:

1. Discovery, identification, and monitoring of nearby sensors: periodically, the M-Hub scans for nearby smart objects announcing their IDs and capabilities. This information about reachable smart objects is kept in the M-Hub database and eventually forwarded to a service of the SDDL cloud.

2. Connecting with sensors: depending on the WPAN protocol used, the M-Hub may first need to establish a communication link with the sensor, over which it will issue synchronous or asynchronous requests to periodically receive the sensor data.

3. Protocol transcoding: Data packets received from the sensors may have different formats and encodings. Thus, the M-Hub must transcode and serialize them before transmitting over the MR-UDP connection. Data transcoding is highly dependent on the type and brand of the sensor device.

4. Caching of probed sensor data: in order to optimize transmission to the cloud over the mobile Internet, the M-Hub may group several sensor data items obtained from nearby sensors...
into a single “bulk message” for transmission to the SDDL cloud. And to do this it must cache the most recent data items probed from the sensors.

5. **Configuring and Probing sensors**: depending on the type of sensor, the M-Hub may eventually or periodically send commands, parameter settings, or request-reply data queries over the WPAN to the connected sensors.

6. **Pre-processing of sensor data**: for some types of sensors, some type of normalization, transformation, or comparison with previous readings are necessary after transcoding and serialization. This data pre-processing should also be done in the M-Hub, pushing to the network edges this basic treatment of sensor data.

7. **Dynamic deployment and life-cycle management of sensor specific modules**: since it is not possible to have built-in modules for every sensor that may be available as the user moves around, the M-Hub provides a run-time deployment and life-cycle management of sensor specific modules.

8. **User privacy**: to ensure the user’s privacy since part of the data collected through sensors are very sensitive (e.g. the user location and the ones provided by a body sensor network, such as physiological data) the M-Hub allows the user to filter which information can be transmitted to the SDDL cloud. Data marked as “private” will only be available locally.

9. **In-network processing**: the M-Hub also provides features that allow developers of AAL applications to distribute the functionalities of their code between the user mobile device and the SDDL processing nodes. For this purpose the M-Hub provides the support for dynamic deployment and management of Java code and CEP (Complex Event Processing) rules. This last one runs in a locally provided Event Processing Agent called MEPA.

10. **Energy-aware processing**: through an energy management component, the M-Hub monitors the mobile device battery level and triggers adaptive actions that adjust its service behavior according to the mobile device energy availability.

### 3.1. Short-Range Sensor, Presence and Actuation API

For uniformly managing the discovery and connection with nearby smart objects (S-OBJ) using different short-range wireless technologies, the M-Hub provides a generic and technology-independent protocol called **Short-range Sensing, Presence & Actuation (S2PA)**. Therefore, the S2PA establishes a common interface (API) for handling the different short-range wireless technologies (WPAN) required by heterogeneous sensors.

The S2PA defines some basic methods and interfaces that unify the handling of different technologies and are responsible for: 1) discovery of, and connection to smart objects, 2) discovery of services provided by each smart object, 3) read and write of service attributes (e.g., sensor values, and actuator commands) and 4) notifications about disconnection of smart objects. For this, S2PA defines the **Technology Interface**. 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. All relevant information regarding smart objects discovery, connectivity, and sensor values obtained from each specific WPAN technology supported is captured through the **TechnologyListener** interface which is implemented by the S2PA service.

As its first realization, we have implemented S2PA for Bluetooth 4.0 (a.k.a. Bluetooth Smart/Low Energy - BLE) and for Classic Bluetooth. In fact, BLE is emerging as a very promising technology, because it is power efficient, enables fast discovery of peripheral devices and supports approximately 2,500 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 smartphone models. Moreover, BLE is being embedded into a growing array of peripheral devices, gadgets, beacons,

---

*Complex Event Processing (CEP) is a low-latency method of tracking and analyzing (processing) streams of information (data) that combines data from multiple sources to infer events or patterns that suggest more complicated circumstances[17].*
small Sensor Tags, etc. Despite its higher energy consumption, the support for Classic Bluetooth (2.0 and 3.0) is also provided since it is usually in a wide range of health and fitness devices, such as the Zephyr BioHarness 3 and Zephyr HxM BT. Moreover, Classic Bluetooth is also used for allowing M-Hubs to communicate between them, which allows the implementation of coordination protocols (e.g. for managing handover of smart objects). Finally, the M-Hub also provides the discovery and connection with any internal sensor available at the mobile device in which it is running, such as the accelerometer, orientation, magnetic field, GPS, battery level, etc.

3.2. The M-Hub Main Components

The M-Hub architecture, illustrated in Figure 2, is multi-threaded and consists of the following local services and managers, all executing in the background and in parallel with user apps.

![M-Hub components diagram](image)

**Figure 2.** M-Hubs components while it interacts with two smart objects, with different WPANs.

The **LocationService** is responsible for sampling the M-Hubs current position and attaching it to whatever message is sent to the gateway, which can be either a static, manually entered geo-point, or the latest geo-coordinate obtained from the smart phones embedded GPS sensor.

The **S2PA Service** implements the TechnologyListener interface and interacts with all nearby smart objects that “talk” the supported WPAN technologies. This service is responsible for the discovery, monitoring and registration of nearby smart objects, 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 smart object, over which the M-Hub interacts in a request-reply mode. Data packets and messages from/to smart objects may have different formats.
and encodings, so it also transcodes sensor data and commands from the specific smart object data format to serialized Java objects, for transmission to the gateways, and vice versa.

Internet messages are received from - and sent to - the SDDL gateways by the **ConnectionService**. This service was implemented on top of the **ClientLib**, a library that provides direct, group and publish-subscribe communication paradigms for mobile nodes. It extends the MR-UDP with mobility-tolerating features, such as handover and Firewall/NAT transversal. These features are transparent to developers, that is, they do not need to handle them. However, ClientLib offers a series of listeners for developers to known when these events occur, e.g., when a node temporarily disconnects from the SDDL cloud. 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. Nevertheless, some messages (e.g. smart object connection/disconnection) have a high delivery priority so that they will be relayed directly to the SDDL cloud, instead of being buffered for further bulk Internet transmission. The ConnectionService also implements a privacy filter, that only forewords to the SDDL cloud the sensor data defined by the user as “public”.

The periodicity and duration of the actions performed by the **LocationService**, **S2PAService** and **ConnectionService** are influenced by the devices current energy level (low, medium, high). This will be set by the **Energy Manager**, which from time to time samples the device battery level and checks if it is connected to a power source.

The **Mobile Client Adaptation Service** is responsible for the instantiation and lifecycle management of dynamically downloaded Java code, used for the deployment of new sensor modules or application defined functionalities. New modules are downloaded from a service running in the SDDL cloud, the Adaptation Manager. The downloaded modules are locally stored in the mobile device at the **Downloaded Module Repository**, in order to avoid the need of resending modules in case of a M-Hub reboot, for instance. In this way, the M-Hub is able to restore its state considering the latest deployed modules.

The **MEPA (Mobile Event Processing Agent) Service** uses an Java-based Esper CEP Engine for Android called Asper [18] to evaluate streams of sensor data looking for certain patterns of data correlations like their time and order of occurrence, expressed as rules. Asper uses a declarative Event Processing Language (EPL), similar to SQL (Structured Query Language), to define CEP rules used to detect the patterns in the streaming data. Simple rules could be: 

\[
\text{SELECT } * \text{ FROM SensorDataTicks WHERE SensorName = "HeartRate", to detect only events of heart rate, or }
\text{SELECT } * \text{ FROM SensorDataTicks (SensorName = "HeartRate") WHERE sensorValue } \geq \text{ 100, to detect only events of heart rate that have a value above or equal to 100 beats per minute,}
\]

where the input data streams are events called SensorDataTicks that represent data received from the smart object. As shown in the Figure 2, the MEPA Service listens to all the messages that are sent from the S2PA Service, which collects data from the smart objects and the smartphone internal sensors. Every time an event is detected, based on the defined CEP rules, it is routed to the ConnectionService to be sent to the SDDL cloud.

4. APPLYING THE M-HUB/SDL SOFTWARE INFRASTRUCTURE TO AAL SYSTEMS

This Section describes how the M-Hub/SDDL software infrastructure can address the challenges related to the development of AAL systems described in Section 1. As previously mentioned, Varshney [3] describes several AAL scenarios: Home Care, Mobile Health, Nursing Home and Assisted Living. We will first address the challenges common to all scenarios, and then discuss the issues that are specific to each one of them. Finally, we present a case study involving the implementation of an AAL system using the proposed software infrastructure.

∥The Esper Engine full documentation can be found at the following URL: [http://www.espertech.com/esper/documentation.php](http://www.espertech.com/esper/documentation.php).
4.1. Challenges Common to All AAL Scenarios

4.1.1. Heterogeneous Technologies The M-Hub/SDDL address the heterogeneity of AAL-related technologies considering two aspects. The first is the heterogeneity of short-range wireless (WPAN) technologies used for the data packets’ transmission from body and environmental sensors to an AAL gateway. To address this challenge, the M-Hub defines the $S2PA API$ that provides a uniform interface for connecting and communicating with smart objects using different WPANs. Although the current implementation of the M-Hub only provides support for Classic Bluetooth and BLE (which are currently the dominant technologies for sensors and actuators employed in health monitoring), its code can be easily extended with support for other short-range wireless technologies.

The second aspect is the heterogeneity of sensor data formats and its encoding. Because it is unfeasible to provide built-in transcoding modules for every possible sensor device that may be discovered, especially in mobile scenarios, it is necessary to deploy these transcoding modules on-demand. To address this challenge, the M-Hub uses the $Mobile Client Adaptation Service$, that allows the download and dynamic deployment of new transcoding modules for newly detected sensors. New application functionality can also be added at runtime through the dynamic loading of executable Java files (jar files) stored in a server of the SDDL cloud. Finally, a life-cycle management of the downloaded code is also provided in the M-Hub, allowing the removal or replacement of modules if they are no longer necessary or have become deprecated.

4.1.2. Reliable Communication Reliability is one of the most difficult challenges to overcome in AAL due to the unstable quality of the wide-area networks, that may employ different wired and wireless technologies (e.g. ADSL, Cellular Networks, Satellites). To address this challenge we built a highly optimized UDP-based communication protocol (MR-UDP) with a small footprint and tangible benefits in regard to communication reliability and Firewall/NAT traversal, providing a robust connectivity solution for wireless health monitoring systems. Making this protocol resilient to IP changes and temporary disconnections are also important capabilities for mobile application environments, where connectivity is usually not reliable.

4.1.3. Power Management It is also important to handle energy consumption in AAL mobile applications and provide means to extend patient monitoring as much as possible until the patient’s mobile device can be recharged. Even though the M-Hub provides many functionalities, its services were carefully developed to consume the smallest possible amount of computational resources. The MR-UDP, for instance, is a lightweight communication protocol that uses a reduced number of connectivity-check packets. In order to reduce the network traffic, the M-Hub also provides the ability to perform some local processing, including Complex Event Processing on the mobile device (In-network Processing).

Nevertheless, the most important feature regarding power management is the provision of an energy management component in the M-Hub that monitors the battery level of the mobile device and triggers adjustments of the M-Hub services according to the available energy. For instance, if the battery level is low, the M-Hub packs the sensor data into a single network package that is sent in predefined time intervals. This time, interval is increased as the available energy decreases. The periodicity of the scan procedure used for discovering nearby smart objects using the short-range wireless technologies is also dynamically adjusted depending on the remaining battery level.

4.1.4. Scalability There is a great demand for AAL systems. According to the US Agency NCHS (National Center for Health Statistics)**, in 2012 there were about 58,500 paid and regulated long-term care services providers that served about 8 million people in the United States. In this context, a single company specialized in patient monitoring can serve a large amount of patients, which may be located in different environments (Nursing Homes, Assisted Living, Home Care and Mobile

**http://www.cdc.gov/nchs/
Health), and share a hardware and software infrastructure. This infrastructure needs to be scalable to prevent a large number of patients requesting services at the same time causing an overload in computing resources, and consequently causing a drop in quality of the monitoring service (e.g. significant increase in user response time).

Another argument in favor of providing high scalability for AAL systems is related to the use of the data generated by this huge amount of patients to detect diseases or other health problems that are most prevalent in the monitored population. Thus, AAL systems can also be used to provide relevant information for the planning of health services and the development of public policies. Therefore, a scalable hardware and software infrastructure capable of storing and processing large volumes of data may be necessary in these cases.

The proposed software infrastructure provides scalability considering three aspects: handling connection with mobile nodes, data stream processing and storage in the cloud. We describe each of these aspects next.

Regarding communication, the infrastructure was designed to allow connections of a great number of mobile devices running the M-Hub and being connected to the SDDL cloud. Scalability is provided by allowing the deployment of new SDDL gateways (see Section 2) on demand and the provision of a dynamic load balancing mechanism for rearranging the amount of connected mobile devices managed by each gateway. SDDL also implements a seamless handover protocol that allows a mobile node (M-Hub) to switch gateways without data loss. Together with MR-UDP resilience to intermittent disconnections, this ensures reliable delivery of messages/data to the cloud, and vice-versa over the mobile Internet connections.

In relation to processing capabilities, the middleware provides a very flexible approach. SDDL can circumvent the processing, memory and storage limitations of mobile devices running the M-Hub by executing, in processing nodes, healthcare-specific workflows or sensor data computations (i.e. both real-time or statistical analyzes) demanding high processing power or large memory. In the SDDL cloud, a processing node can either analyze data streams related to a single patient allowing it to infer his/her activity, behavior and health condition; or else, it could process aggregations of sensor data streams generated by many patients, what would allow inferences related to a given population that usually has a similar health condition or chronic disease such as asthma. As the demand for data processing in the SDDL cloud increases, new processing nodes can be dynamically added to the infrastructure without degrading much the overall system’s performance. Nevertheless, the workload of processing nodes can become unbalanced. To overcome this issue, a load-balancing mechanism is provided, as described in Section 2. Regarding the processing capabilities, another point to consider is that if all data processing is delegated to the cloud, this could be very expensive in terms of energy and network bandwidth to the mobile device running the M-Hub, since frequent communication would be necessary. In order to avoid this problem, the M-Hub can perform local computations implemented either in plain Java, or expressed as CEP rules. This allows pre-processing of sensor data by performing operations such as summarization, filtering, data transformations, or the inference of higher-level context information.

Finally, regarding storage capabilities, the M-Hub can use the SDDL cloud for storing and managing information about patients, which can be accessed by authorized health professionals or family members from any device.

4.2. Challenges Concerning AAL Specific Scenarios

4.2.1. Home Care and Mobile Health Scenarios Home Care is a continuous mode of health service delivery aimed at the continuity of hospital care at home, carried out by a multidisciplinary healthcare team. Home Care prevents prolonged hospital stays by allowing the continuity of the patient care at home. Usually, patients in Home Care are moderately independent (maybe even live alone), and receive visits of a health professional on a regular basis, or as needed. In this case, the health professionals are not always at the patient’s home location but are likely to be in the same region (e.g. city or district). Even though the patients’ cognitive and physical abilities may be somehow limited, they are able to interact with the health monitoring system without major problems. The location of patients is often restricted to a single place - their home. In the
Mobile Health scenario, patients are also able to move to neighboring places (medium mobility). In this case, patients’ independence, mobility, and physical and cognitive abilities are higher than in the Home Care scenario. Therefore, they can have a high degree of interaction with the health monitoring system. Because of their increased mobility, they can also move around the city. In this case, the healthcare professionals are also probably farther away, but still within the same metropolitan area or region.

As already mentioned, each M-Hub is also able to connect and communicate with several peripheral/sensor devices simultaneously, and through different short-range wireless communication technologies. These sensor devices may constitute the patient’s body sensor network or may be environmental sensors spread in the places where the patient lives and moves around. In both cases, the M-Hub can collect and analyze the patient’s vital signs or environment data (e.g., high ambient temperature, gas leakage, etc.) in order to detect any anomaly. In this case, an alert can be issued to healthcare professionals and family members through SDDL.

Specifically in the Home Care scenario, the M-Hub can be used to send notifications (using a wide-range wireless network) to neighbors or family members who live near the patient, so that they can give a primary assistance. In case of an emergency, the patient’s location is very important in order to guide the healthcare professionals (or rescue team) to the correct place where the patient is. If the patient is unconscious or has other problems that prevent him/her from telling his/her current location, healthcare professionals can infer that the patient approximate location corresponds to the location of the home sensors with which the M-Hub is currently connected (i.e., indoor positioning by sensor proximity). If the patient is an outdoor environment, the location provided by the M-Hub’s GPS sensor can be used.

In the Home Care scenario, the healthcare monitoring system usually uses a WLAN (WiFi) for transmission of data and alerts, but 3G/4G technologies (WWAN) may be also employed. In Mobile Health scenarios, WLANs are still widely used but since the patient has more mobility, it is expected that 3G/4G networks are used when the patient is far from its WiFi access points. Because the M-Hub uses SDDL’s MR-UDP as its communication protocol, it can reliably transfer data over both WLAN and WWAN technologies with seamless transition between both types of networks. This reliable transmission of sensor data to health professionals is particularly important in AAL. For instance, if an emergency alert is not delivered on time, health professionals will not be able to arrive timely in order to answer a critical health situation. In Mobile Health scenarios, on the other hand, it is very likely that the patient’s M-Hub suffers some disconnections while moving. To circumvent this, MR-UDP also supports reliable delivery of data in situations where connectivity between mobile devices (e.g. the M-Hub) and the SDDL cloud is weak or intermittent.

Beyond serving as an intermediary for body sensors, the M-Hub can also interact with ambient sensors in order to detect conditions related to the patient’s surrounding environment (at home, at work, at school or on the street) that are unfavorable to the patient’s health and/or that may put his/her life at risk. For instance at home, when the patient approaches the kitchen, the M-Hub can interact with the gas sensors in order to detect any leakage. Lighting sensors can also help to detect changes in the patient’s daily routine. For instance, if during the day the patient stays many consecutive hours in a room without luminosity (natural or artificial) this information can evidence that the patient has a problem. Or else, if the room light is turned on many times during night hours, this may indicate that the patient is sleepless or anxious. In outdoor environments, sensors scattered throughout the city, measuring temperature, humidity, carbon monoxide concentration, and other gasses can be used to determine whether the climatic and atmospheric conditions are detrimental to the patient health.

While transiting through different places, the monitored patient may encounter various sensors distributed and embedded throughout the environment. It is possible that the M-Hub does not have all the required modules installed locally in order to connect to - and receive data from - all the sensors found. The M-Hub Mobile Client Adaptation Service can check with the Adaptation Manager that runs on a server in the SDDL cloud if there is any module available that is compatible with the discovered sensors. If this is the case, they can be downloaded and installed dynamically. This is necessary only at the first time a new sensor is discovered.
The time required to discover sensors and/or actuator services, connect and receive data from them is very important. A long delay in these operations may cause harmful consequences. In the Home Care scenario, for instance, if the time to connect and receive data from the gas sensor is slow, the time to detect a leakage is also affected, putting the patient’s life at risk. While the delay will largely depend on the short-range wireless technology being used, the M-Hub was designed to discover and connect to smart objects very quickly (see the evaluation presented in Section 5).

In the Mobile Health and Home Care scenarios, since patients are fairly independent, it is likely that in some situations they may want to withhold certain information with the aim to preserve their privacy. For instance, the patient may not want to expose his/her location in certain places. The M-Hub provides a filter component that allows the user to enable and disable the collection and transmission of some data gathered from any sensor. Developers of health monitoring applications can use this feature to give the user the ability to explicitly determine when and where he/she wants to disclose his/her context (i.e., its sensor) data.

4.2.2. Nursing Home and Assisted Living Scenarios In many situations, elderly people are unable to live alone and must have continuous and long-term care services. In Nursing Home and Assisted Living (also called Residential Care in some countries), the patients’ autonomy and degree of physical and cognitive abilities are more limited than in the Mobile Health and Home Care scenarios. In Nursing Homes, patients are entirely dependent on a nurse or healthcare staff. These professionals are usually quite close to the patients (e.g., less than 200 meters). In many cases, the physical and cognitive abilities of patients are also very limited, and they are not able to interact with the health monitoring system. Patient mobility is very low and restricted to the Nursing Home. In Assisted Living, patients have some degree of autonomy to do most of their daily activities but receive support from staff for some activities, such as cleaning and cooking. The caregivers are somewhat close to patients (e.g., in the same neighborhood). The physical or cognitive abilities of these patients are less limited than the other scenarios, and sometimes they may be able to use the health monitoring system. We can say that the patient’s mobility is at medium scale since he/she can move around in a restricted space (e.g., within the Assisted Living space or a region close to his/her home).

As in the Home Care scenario, the M-Hub can be used to collect information from body sensors and ambient sensors installed in the rooms of the Nursing Home or Assisted Living complex and the collected sensor data and eventual emergency alerts can be sent via SDDL to the healthcare staff. However, considering that healthcare professionals will usually be close to the patient, there are other possibilities for using the M-Hub functionality. For instance, if both the patient and the caregiver carry a smartphone executing the M-Hub, the patient’s M-Hub can send the sensor data directly to the caregiver’s device, either through the short-range WPAN technology or through SDDL, using the WiFi network. Or else, if only the caregiver runs the M-Hub on his/her smartphone and is also close to the patient, then the patient’s body sensors and environment sensors can be discovered and read by the caregiver’s M-Hub. In these scenarios, due to the somewhat limited physical and cognitive abilities of patients, perhaps operations such as discovery and connection to (body and environment) sensors and actuators can be executed without the patient’s intervention. The M-Hub was designed to discover and connect to smart objects/sensor devices autonomously, and use any available wireless Internet connection, enabling its use in scenarios where patients or caregivers have no ability or time to interact with the health monitoring system.

4.3. Case Study: Mobile Activity and Intensity Recording System

One important class of AAL systems is called Human Activity Recognition (HAR), whose task is to recognize patterns of human movement activity (e.g., walking, running, sitting) from various types of low-level sensor data. This section presents a case study involving the implementation of a HAR using the proposed software infrastructure. This AAL system is called Mobile Activity and Intensity Recording System (MAIRS) [20], and was designed to be executed in personal mobile devices running the M-Hub middleware connected to the SDDL cloud. MAIRS was developed...
by the Intelligent Distributed Systems Laboratory (LSDi)†† at the Federal University of Maranhão (UFMA) in collaboration with the UFMA University Hospital (HUUFMA) in order to support the monitoring of patients with chronic diseases.

Movement activity recognition is desirable in several types of treatment of chronic diseases, especially for heart (e.g. hypertension, heart failure, atrial fibrillation) and respiratory (e.g. chronic bronchitis, emphysema, asthma) problems. It allows finding out if the patient is doing the physical activity routine recommended by health professionals. For example, it is possible to infer whether the patient walks or runs frequently or if he/she has a more sedentary lifestyle. A common approach used for classifying activities based on sensor data that can be related to the body movement is based on the use of machine learning algorithms. The choice of which technique is better suited for a given application may depend on the set of activities to be inferred, the available computational resources for running the algorithm and the size of the base training sets.

In some situations, it is important to check how the patient is responding to the performed physical activities, and detect whether the level of effort is compatible with the patient’s current physical capacity, imposed by her/his chronic condition, age, weight and other factors. This is called intensity measurement or measurement of body stress. If the values measured are not following the recommended intensity patterns, the activity detection system must decide the actions to be taken, such as to issue a warning to the health professional responsible for the patient, or ask the patient to increase or decrease the intensity of the physical activity being performed in order to better suit the established limits.

By using the M-Hub, MAIRS allows data gathering from sensors usually available on personal mobile devices (internal sensors), as well as from external sensors, including both wearable health sensors and ambient sensors, i.e. sensors collecting data from the patient's environment. In particular, MAIRS uses accelerometer data for inferring the patient’s performed activity and her/his heart rate for computing its intensity. MAIRS is currently able to recognize the following movement activities: walking, running, jumping, standing, lying down, walking up and down stairs. The activity intensity is related to the physiological effects (e.g. fatigue, stress, muscle fatigue, lactic acid) of the activity being performed, considering the health condition and the physical limits of each patient. The intensity is usually mapped to a scale comprising a set of ranges called intensity zones[21].

The requirements established for MAIRS development were: 1) Support for sensors heterogeneity: it should be able to interact with different types of sensors (wearable, portable or embedded in smart environments), making it possible to obtain different types of information about the patient; 2) Activity Recognition: it should be able to group, process and classify the obtained sensor data automatically and in real time; 3) Intensity Measurement: it should be able to gauge the intensity of the user performed movement activities; 4) Scalable Data Processing in the cloud: tasks such as activity recognition and intensity measurement should be performed in the cloud, relieving the use of the mobile system resources; 5) Reliable communication: communication between the mobile device and the cloud requires reliable delivery of data. Since the patient may be moving, the mobile device may possibly undergo periods of weak or intermittent connection. Thus, the monitoring system needs to store data locally for further bulk Internet transmission.

Figure 3 illustrates the MAIRS architecture, showing the main components that are responsible for achieving the established requirements. Because it is an AAL distributed system, some MAIRS components run on the user mobile device (S2PA, SPS, and CS), while others run on a processing node at the SDDL cloud (HURS and IMS).

Short-Range Sensor, Presence and Actuation API Service (S2PA) is provided by the M-Hub (see Section 3). It is responsible for the discovery, connection, and raw data acquisition from internal and external sensors. Considering MAIRS specific needs, S2PA was configured to enable only the accelerometer and heart rate sensors used for inferring the user activity and its intensity, respectively, as well as the GPS, to obtain the patient location when performing movement activities.

††http://www.lsdi.ufma.br
Signal Preprocessing Service (SPS) is the component responsible for raw data preprocessing, transforming it into the input format required by the classifier. Preprocessing takes place at the mobile node and is performed every 2.58 seconds in the bulk of accelerometer data collected at a frequency of 50Hz. The preprocessing result is a reduced set of numeric values that represent characteristics of the accelerometer data: the average values of each axis (X, Y, Z), as well as the square root of the average obtained from the medium for each axis. These values are used as input for the classifier responsible for the user activity recognition, which is performed at the SDDL cloud.

Connection Service (CS) is also a component provided by the M-Hub (Section 3). It is responsible for establishing connections through 3G/4G or Wi-Fi networks, allowing mobile clients to send the preprocessed sensor data to the SDDL cloud through a gateway. In case of loss of connectivity to the SDDL cloud, the MR-UDP protocol (Section 3.2) allows the system to store the preprocessed data in a buffer on the mobile device, forwarding it when the connection to the gateway is reestablished.

Human Activity Recognition Service (HURS) is the component responsible for carrying out the recognition of the activity being performed by the user. When the gateway receives the preprocessed data sent by the mobile client, it publishes this data into a specific topic in the DDS domain that is subscribed by HURS. The preprocessed data is used as input to a machine learning algorithm, a previously trained classifier, which is responsible for categorizing the activity that the user is performing. The specific machine learning algorithm used in HURS is IBk [22], available in the WEKA library [22].

Intensity Measurement Service (IMS) is a component responsible for measuring the intensity of the user movement activity inferred by HURS. For this reason, IMS and HURS are executed together in the processing nodes. In general, the computation of the activity intensity can take into consideration several parameters, including the ECG, heart rate and the volume of oxygen...
consumed. The choice of which are the best parameters to be used depends on the monitoring requirements, such as the intended monitoring environment and the availability, degree of intrusiveness and portability of the sensors that will be used. IMS adopts the dominant methodology found in the literature for measuring the movement activity intensity that is based on a periodic analysis of the patient heart rate [23]. The closer it is to the patient maximum heart rate, the greater will be the activity intensity measurement.

It is possible to dynamically add and remove processing nodes depending on the system load (amount of requests for activity inference from mobile clients), providing computational elasticity. The SDDL cloud maintains a database related to the patient that stores all detected activities and their intensities, which are enhanced with context meta-data such as the inference timestamp and the user geographic location when the activity recognition was performed. The database can be accessed remotely by physicians, caregivers and family members in a controlled manner (considering the user privacy requirements) through a Web system.

Considering that the M-Hub and SDDL components satisfactorily met the requirements for the development of MAIRS, one can argue that the proposed software infrastructure is adequate for the development of AAL systems focused on the human activities recognition, as well as others patient monitoring systems that have similar requirements, such as the support for sensors heterogeneity, reliable communication, and scalability.

5. CURRENT STATUS AND EVALUATION RESULTS

5.1. M-Hub Evaluation: Discovery and Connection with Smart Objects

As described in Section 4, in almost all AAL scenarios it is very important to discover, connect and receive data from sensors and actuators automatically and in a fast and opportunistic way. As mentioned, serious consequences may arise if the time for performing any of these operations is high. For example, in some scenarios such as Mobile Health, if the discovery and connection to smart objects is not sufficiently agile, it may become impossible to receive data from ambient sensors in places/rooms where the patient remains only for a short period of time. This section presents and discusses the results of experiments aimed to evaluate some aspects of the interaction between the M-Hub (acting as a residential gateway) and heterogeneous smart objects (sensor tags and wearable devices) close to him in scenarios using different short-range wireless communication technologies (BLE and Classic Bluetooth). In these experiments, we measured the time that the M-Hub takes to discover the sensors around it, the time it takes to connect to these devices, and the time until it begins to receive raw data collected by the sensors. Therefore, the experiments in this section only evaluate the communication between the M-Hub and sensors, without the intent to evaluate any aspect related to the communication between the M-Hub and the SDDL cloud.

For the performance evaluation of the M-Hub using BLE enabled-sensors, we used four Texas Instruments SensorTags [24]. M-Hub’s S2PA Service was configured to perform a BLE device scan every 3 seconds, and the scan duration was set to 2 seconds. For running the M-Hub we used a Motorola Moto X smartphone executing Android 4.4.2 KitKat. The evaluation metrics were: Connection Time (CoT); Service Discovery Time (SDT); Time to Receive the first data (TR); and Total Time to Receive the first data (TTR), all measured in seconds. We collected these parameters both for the first connection of the M-Hub with the SensorTags, and for follow-up reconnections. The TTR is the sum of CoT, SDT, and TR, and corresponds to the total time required for receiving the first data from the sensors of a SensorTag since the M-Hub discovered it. 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 Android KitKat in regard to BLE is not optimized. For example, connections to each of the sensor devices are done only sequentially, while BLE would allow parallel/concurrent connection operations. The obtained results of our experiments are presented in Table I.

Table I. M-Hub Performance: Bluetooth Low Energy Sensor

<table>
<thead>
<tr>
<th></th>
<th>CoT (s)</th>
<th>SDT (s)</th>
<th>TR</th>
<th>TTR (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mean value</strong></td>
<td>0.21192</td>
<td>9.132</td>
<td>0.64958</td>
<td>9.9935</td>
</tr>
<tr>
<td><strong>Std. Deviation</strong></td>
<td>0.1578</td>
<td>0.14381</td>
<td>0.1511</td>
<td></td>
</tr>
</tbody>
</table>

For BLE technology-enabled sensors, the slowest operation is the services discovery, in which the M-Hub retrieves all the services, characteristics, and descriptors that a sensor device possesses, according to the GATT protocol. So, the latency of this operation will be proportional to the number of services of the sensor device, which in the case of the SensorTags is six, due to its six sensors. However, this services discovery time decreases very much for follow-up reconnections, as can be seen from Table I.

For the evaluation of the M-Hub accessing sensors with Classic Bluetooth technology, we used the Zephyr BioHarness 3. The configuration parameters of S2PA and the smartphone used in these experiments are the same as in the previous experiment for BLE technology sensors. For the experiment, the BioHarness 3 sensor was already paired with the smartphone before the connection step. Here, the considered metrics were Connection Time (CoT); Time to Receive the first data (TR); and Total Time to Receive the first data (TTR), all of them measured in seconds. The service discovery time was not measured because for some sensors, such as the Zephyr BioHarness 3, there is not a service discovery phase. Instead, the M-Hub must already be aware of the services that this sensor device provides. To receive the data, it simply sends an Enable request. If the sensor does not implement the requested service, the M-Hub will simply not receive any data. However, other Classic Bluetooth sensors, such as the Zephyr HxM BT, do not operate in this way: the sensor simply starts sending all available data as soon as the Bluetooth connection is established and also does not require a service discovery phase. Again, we ran each experiment 12 times and calculated the mean value and the standard deviation. The results are presented in Table II.

Table II. M-Hub Performance: Classic Bluetooth Sensors

<table>
<thead>
<tr>
<th></th>
<th>CoT (s)</th>
<th>TR</th>
<th>TTR (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mean value</strong></td>
<td>1.39415</td>
<td>0.21756</td>
<td>1.61171</td>
</tr>
<tr>
<td><strong>Std. Deviation</strong></td>
<td>0.29737</td>
<td>0.31839</td>
<td></td>
</tr>
</tbody>
</table>

The experiments show that M-Hub’s Total Time to Receive Data (TTR) is low for both technologies: \( \approx 1 \) second for BLE sensors for follow-up reconnections and \( \approx 1.6 \) seconds for Classic Bluetooth sensors. The highest TTR (\( \approx 10 \) seconds) is observed only in the first connection to BLE sensors, i.e. when the services provided by the BLE sensor are unknown. In indoor environments, e.g. the patient’s home, in the patient’s home or in a hospital, where patients may walk around entering and leaving the same rooms several times, the \( \approx 1 \) and \( \approx 1.6 \) seconds delay for receiving data or detecting the presence of environment sensors is sufficient for detecting where the patient is located and reacting to critical events, such as a gas leakage, smoke, sudden walking movement interruption (e.g. the patient faint and fell down, etc.), as discussed in Section 4. For outdoor environments, where the patient’s M-Hub should opportunistically gather data from potentially unknown sensors, even the \( \approx 10 \) seconds delay imposed by the first-time

---

discovery of BLE is sufficient for obtaining data from nearby sensors considering that the patient’s movement speed is only \( \approx 5 \text{km/h} \) (when walking) or at most \( \approx 10 \text{km/h} \) (riding a bicycle). This holds true even considering Bluetooth devices with only a small range of \( \approx 50 \text{m} \), while the Bluetooth specification considers a \( 100 \text{m} \) range for outdoor environments. Therefore, we believe that the M-Hub and our proposed software infrastructure are able to deal with the automatic and opportunistic sensor discovery and data gathering requirements of all the AAL scenarios described previously.

5.2. SDDL Scalability Evaluation

In Section 4.1.4, it was argued that AAL systems should be scalable. Here, we present performance results that demonstrate the SDDL scalability regarding the amount of connected mobile nodes requesting processing of context data in the SDDL cloud. To make possible the scalability evaluation of the SDDL considering AAL scenarios, we used an modified version of the MAIRS (see Section 4.3) and a client simulator, a program that launches an arbitrary number of concurrent clients that connect to a group of SDDL gateways. The simulated mobile clients run on a Java platform for personal computers and workstations, instead of the Android operating system, used by actual mobile clients.

It is also important to note that both types of mobile clients (simulated and actual) use the same protocol for communication with the SDDL cloud: the MR-UDP (see section 3.2). In this way, the SDDL gateways treat the connections of simulated mobile clients in exactly the same way as it would treat the actual M-Hub device connections. The workload submitted to the SDDL cloud would also be analogous. Thus, the results obtained by using a simulation technique are equal or very close to what would be observed in the case of using real mobile devices with the M-Hub installed. One of the main reasons that led us to choose the simulation approach is due to the need to perform the experiments in a controlled environment, with the ability to repeat these experiments several times, when necessary. Also, to run a similar experiment using real mobile devices, it would be needed thousands of smartphones and/or tablets, since the intention was to evaluate the SDDL cloud considering a large scale of clients, which was not feasible.

To simplify the experiment, the processing nodes in the SDDL cloud only performs the movement activity recognition, not performing other MAIRS functions such as intensity measuring and the storage of detected activities into a database. The simulation works as follows. After initialization, each simulated mobile client sends a request for activity recognition to the SDDL cloud every 2.58 seconds over a period of 60 seconds. The sensor data contained in each request is obtained randomly from a trace file containing 1,066 instances of already preprocessed acceleration data. In the SDDL cloud, the gateway to which the client is connected publishes the received data in the DDS domain in order to enable one or more processing nodes running the HURS component (see Section 4.3) on the same domain to receive the data. For each received request, a new thread is instantiated for running the activity recognition task based on the received data. After performing the movement activity recognition, the thread publishes a reply message containing the inferred movement activity on another topic in the DDS domain, to which the gateway is subscribed. The gateway will then forward the inferred movement category to the client that made the request.

In order to evaluate the SDDL scalability, we increased the number of clients and measured if the SDDL middleware was able to maintain the quality of service when adding new gateways and processing nodes to the cloud infrastructure. The used metric to measure the quality of the activity recognition service was the average response time. The response time of each request is calculated as the difference between the time a response arrives at the client and the instant of time when the request was sent to the SDDL cloud. We performed the described experiment in three scenarios involving different numbers of clients. Starting with 100 clients each subsequent scenario was increased by a factor of 10. By adding new gateways, it was expected that SDDL would be able to reduce the communication delay and by adding new processing nodes (orchestrated by the load balancing mechanism) we expected a reduction of the processing time.

The evaluation was performed using a cluster located at the Intelligent Distributed Systems Laboratory (LSDi) of the Federal University of Maranhão (UFMA). The experiments were
performed with the following hardware configurations: gateways and processing nodes were running in virtual machines with the Ubuntu Linux 14.04 operating system and 1 Gigabyte of RAM. These virtual machines were running on Intel Core i7 machines, with 16 Gigabytes of RAM, installed with Ubuntu Linux 14.04. The clients were running in physical machines equipped with Intel Core i7 processor and 12 Gigabytes of RAM, also running Ubuntu Linux 04.14 LTS. The number of virtual and physical machines varied according to each simulated scenario, as specified next.

- First scenario: One group containing 100 clients running on a physical machine, sending requests to a single gateway that, in turn, transmitted the request to a single processing node running on the same virtual machine as the gateway.
- Second scenario: Two groups containing 500 clients. Each group was executed on a different physical machine, sending requests to two gateways and two processing nodes. In this case, two virtual machines were used in the SDDL cloud, each one running a processing node and a gateway. Both virtual machines were executed on a single physical machine.
- Third scenario: Four groups containing 2,500 clients (totaling 10,000 clients). Each group was running on a different physical machine, sending requests to 10 gateways and 10 processing nodes. In this case, 10 virtual machines were used, each one containing a processing node and a gateway. These virtual machines were running on two physical machines, with five virtual machines each.

Table III summarizes the obtained results. The experiment of each scenario was executed five times (corresponding to one series). The average response time (Round Trip Delay) for each scenario corresponds to the average of the average response times obtained in each series.

Table III. Average activity recognition response times with variable number of mobile clients, gateways and processing nodes.

<table>
<thead>
<tr>
<th>Clients</th>
<th>Gateways</th>
<th>Processing Nodes</th>
<th>Average Response Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1</td>
<td>1</td>
<td>0.01449</td>
</tr>
<tr>
<td>1,000</td>
<td>2</td>
<td>2</td>
<td>0.02289</td>
</tr>
<tr>
<td>10,000</td>
<td>10</td>
<td>10</td>
<td>0.46961</td>
</tr>
</tbody>
</table>

In addition, to show the performance of the SDDL in scenarios where the dynamic resource allocation mechanisms and load balancing is not used, the Table IV presents the evaluation results where the number of clients varies, but the amount of computational resources remains constant.

Table IV. Average activity recognition response times with variable number of mobile clients and fixed number of gateways and processing nodes

<table>
<thead>
<tr>
<th>Clients</th>
<th>Gateways</th>
<th>Processing Nodes</th>
<th>Average Response Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1</td>
<td>1</td>
<td>0.01449</td>
</tr>
<tr>
<td>1,000</td>
<td>1</td>
<td>1</td>
<td>0.06866</td>
</tr>
<tr>
<td>10,000</td>
<td>1</td>
<td>1</td>
<td>5.99341</td>
</tr>
</tbody>
</table>

Analyzing the obtained results in the first and second scenarios (100 and 1,000 clients respectively) in Table III, we observe a very small increase in the average response time that is almost negligible. However, when we compare the first two scenarios with the third one (10,000 clients) in the same table, we observe a slightly larger difference in the average response time, even though it is still less than 0.5 second. We attribute this slightly higher response time to the communication overhead imposed by the addition of many processing nodes and gateways to the SDDL cloud infrastructure, whose nodes communicate using DDS (see Section 2). However, we consider that this difference is quite acceptable considering the amount of simulated clients (10,000).
Taking, for instance, a monitoring scenario of patients with high mobility, a response time of 0.5 second (or even a little bit larger) seems good enough to provide for healthcare professionals and patients the feeling that the activity recognition is being done in near real time.

Considering also the results of Table IV, we conclude that the dynamic allocation mechanism of computational resources of the SDDL cloud, used in the previous experiment, was largely responsible for maintaining the system scalability. This is due to the fact that the load balancing between both gateways and processing nodes prevents the increase in the average response time as the number of clients increases. For example, the average response time for clients in scenarios with 1,000 clients for 2 gateways and 2 processing nodes is about 3 times less than when we used only 1 gateway and 1 processing node for the same number of clients. The average response time in the scenario with 10,000 clients for 10 gateways and 10 processing nodes is approximately 6 times lower than when we used just 1 gateway and 1 processing node for that same number of clients. These results reinforce the conclusion that SDDL provides the required computational capability to execute large-scale AAL systems.

6. RELATED WORK

In the literature, one can find several proposals for health monitoring in AAL, but few consider the combined use of IoT and cloud computing. In this section, we will describe briefly some of these proposals, comparing them with the M-Hub/SDDL software infrastructure. For comparison, we analyze the characteristics of each related work that allow addressing some of the challenges described in Sections 1 and 4: Comprehensiveness of Scenarios, Reliable Communication, Heterogeneous Technologies, Scalability and Power Management.

Rahmani et al. [24] present the Smart e-Health Gateway, a residential gateway designed to be installed in hospitals and smart homes that supports different communication protocols and acts as a bridge between the body and environmental sensors and the Internet. It receives data from different WPANs technologies (e.g. ZigBee, 6LoWPAN, Bluetooth, Wi-Fi), performs protocol conversion, and provides other higher level services such as data aggregation, filtering, and dimensionality reduction through a Local Data Processing Module. To send and receive data to/from the cloud, the Smart e-Health Gateway can use different communication technologies, such as 3G, LTE, WiFi, and Ethernet. The Smart e-Health Gateway was designed to be easily reconfigured to support more protocols and standards. The Local Data Processing Module is also capable of executing different data mining and machine learning algorithms proving support for in-network processing. Moreover, the Smart e-Health Gateway can act as an intermediate storage for certain time period, allowing to circumvent an eventual network unavailability. Once the Internet connection is recovered, the data is sent to the cloud. The cloud computing platform provides broadcasting, data warehousing, and Big Data analysis. A graphical user interface for data visualization is also provided.

Cubo et al. [25] present a cloud-based IoT platform for AAL. The main goal of this system is to assist patients and healthcare professionals by providing a health monitoring system with support for a large number of patients. The solution is designed to attend the scenario where the patient can be at home or at a hospital, being monitored by several sensors. All sensors communicate using IEEE 802.15 with the AAL gateway that is implemented using a Raspberry Pi board. This gateway uses a local area network in order to send via TCP/IP the gathered sensor data to an application that suggest plausible diagnoses. This application runs at the Google Cloud Platform and performs data analysis using BigQuery, the Google large-scale data analytics.

Balasubramanian and Stranieri [26] present a cloud-based architecture known as AppA (Assistive Patient monitoring cloud Platform for Active healthcare applications). The scalability of the solution is justified by the ability to add new instances of servers to the cloud infrastructure. Each server is used to process data from a subset of patients who use the services provided by the AppA. The idea is to provide a system where the hospital can hire only the services it needs to meet each patient-specific treatment requirements (data processing, storage, processing functions for specific diseases etc.). A Samsung Galaxy Tab 2 was used as the AAL gateway in this work. The monitoring application was built using Android SDK. The gateway connects to the employed sensors using
Classic Bluetooth. The focus of this work is on fall detection and knee rehabilitation of elderly patients. Therefore, the AAL gateway communicates with three types of sensors: Accelerometer, Electromyogram (EMG) and Electrocardiogram (ECG), because these sensors data are used by the fall detection algorithms. The accelerometer data is useful for knee rehabilitation monitoring.

Yang et al. [27] present the iHome-Health, a home-based platform that uses the IMedBox (Intelligent Medicine Box) residential gateway, which is equipped with a high-performance tablet PC. Sensors can be connected to the iMedBox through several network technologies, such as Ethernet, RFID, Zigbee, WiFi, Bluetooth, and 3G/4G networks. All the collected data from sensors are interpreted, stored, and displayed locally in the iMedBox. The collected data can also be forwarded to the cloud for the clinical diagnosis or further complex analysis.

Reddy et al. [28] present a web-based telemedicine/healthcare cloud support for connecting high complexity hospitals (that provide very specialized treatments), community health centers and emergency care units, with the aim to provide access to remote areas. The proposed system is called CHMS (Cloud Framework for Health Care Monitoring System). In short, the steps involved in the CHMS usage model consist of: 1) Patients data are collected using body sensors networks and are periodically sent to an AAL gateway running on a personal mobile device, called acquisition module. Sensors collect data like ECG, Glucose etc. for further analysis; 2) Data are sent by the AAL gateway to the cloud using Wifi or cellular network. 3) In the cloud, there are services that can apply some type of processing and analysis on the data received and can perform immediate actions based on the results, such as alerting ambulances and healthcare professionals. The cloud forwards the analyzed data to the patients mobile device. CHMS enables dynamic allocation of virtualized servers in the cloud in order to meet an increase in the demand for services or to replace failed servers.

Doukas et al. [29] present a cloud-based IoT system that manages the data collected by medical and environmental sensors which are forwarded to a residential gateway called IoT Dragino Gateway. This gateway is equipped with a Bluetooth communication module for gathering data from the sensors and uses a WLAN to forward them to the cloud where applications, like a medical record system or an emergency detection platform, are executed. A web-based application that displays in real-time the monitoring data is provided. Dragino IoT Gateway is a stationary hardware designed to be installed indoor.

6.1. Discussions

Table V summarizes a comparative analysis of the related work, including the M-Hub/SDDL.

Regarding the comprehensiveness of AAL scenarios, one can see that only the proposed software infrastructure (M-Hub/SDDL) and the work of Reddy et al. [28] can be applied in all AAL scenarios described in Section 4. This holds for the work of Reddy et al., since they use gateways that run on personal mobile devices (e.g. tablets and smartphones), which can be carried anywhere by the patient.

In respect to reliable communication, Reddy et al. [28], Doukas et al. [29], Cubo et al. [25] and Rahmani et al. [24] use TCP/IP as the communication protocol between the gateway and the cloud, which provides mechanisms to ensure reliability in message delivery. Additionally, the AAL gateway proposed by Rahmani et al. [24] has a built-in support for circumventing eventual network disconnections by using a local message repository which temporarily keeps a copy of the data collected from patients until they can be delivered to the cloud. However, the gateway is fixed and requires an interruptible power supply, limiting the patient monitoring to environments such as hospitals or smart homes. In the M-Hub/SDDL, the protocol used for communication between the gateway and the cloud is the MR-UDP. This protocol besides reliable message delivery and support network disconnections also supports firewall and NAT traversal and IP address changes, improving patient mobility. The work of Balasubramanian and Stranieri [26] and Yang et al. [27] do not mention what are the communication protocols used.

Table V. Comparison between M-Hub/SDDL and Related Work

<table>
<thead>
<tr>
<th>Work</th>
<th>Comprehensiveness of scenarios</th>
<th>Reliable communication</th>
<th>Heterogeneous technologies</th>
<th>Scalability</th>
<th>Power Management</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rahmani et al. [24]</td>
<td>Hospital and smart homes</td>
<td>TCP/IP + support for network disconnections based on temporary local storage</td>
<td>ZigBee, 6LoWPAN, Bluetooth, and Wi-Fi. Can be extended</td>
<td>cloud</td>
<td>Uniformed</td>
</tr>
<tr>
<td>Cubo et al. [25]</td>
<td>Home and hospital</td>
<td>TCP/IP</td>
<td>IEEE 802.15 standard</td>
<td>cloud + add computing resources on demand</td>
<td>Uniformed</td>
</tr>
<tr>
<td>Balasubramanam and Stranieri [26]</td>
<td>Hospital</td>
<td>Classic Bluetooth</td>
<td>cloud + add computing resources on demand</td>
<td>Uniformed</td>
<td></td>
</tr>
<tr>
<td>Yang et al. [27]</td>
<td>Home</td>
<td>Uniformed</td>
<td>RFID, ZigBee, Wi-Fi, and Classic Bluetooth</td>
<td>Uniformed</td>
<td></td>
</tr>
<tr>
<td>Reddy et al. [28]</td>
<td>All scenarios</td>
<td>TCP/IP</td>
<td>Uniformed</td>
<td>cloud + add computing resources on demand</td>
<td>Uniformed</td>
</tr>
<tr>
<td>Doukas et al. [29]</td>
<td>Home</td>
<td>TCP/IP</td>
<td>Classic Bluetooth</td>
<td>Uniformed</td>
<td>Uniformed</td>
</tr>
<tr>
<td>M-Hub/SDDL</td>
<td>All scenarios</td>
<td>MR-UDP + support for network disconnections, firewall and NAT traversal and the support for IP address changes</td>
<td>Classic Bluetooth and Bluetooth LE. Can be extended. Download of sensor modules from the SDDL cloud</td>
<td>cloud + add computing resources on demand + load balancing</td>
<td>Energy Manager, CEP and MR-UDP</td>
</tr>
</tbody>
</table>

Regarding the heterogeneity of technologies, the work supporting the widest range of WPANs technologies is Rahmani et al. [24] (ZigBee, 6LoWPAN, Bluetooth, Wi-Fi). In this work, it is mentioned that other technologies and standards can be incorporated into the AAL gateway. Currently, M-Hub implements the support for Classic Bluetooth and BLE technologies that are present in a wide range of sensor devices. M-Hub provides a uniform interface for handling different WPAN sensors technologies and its code was designed to be easily extended for providing support for other technologies, as needed. The others related work do not mention if the support for other short-range wireless communication technologies could be added besides the ones that are already provided. In addition, M-Hub is the only AAL gateway that allows the dynamic loading of sensor modules. This favors the use of our software infrastructure in mobility scenarios, since as the patient moves his/her device may discover and interact with new sensors that were not known in advance.

Concerning scalability, although all described approaches use a cloud computing platform to perform computationally intensive tasks, few mentioned specific mechanisms to ensure scalability. Cubo et al. [25], Reddy et al. [28], Balasubramanam and Stranieri [26] and M-Hub/SDDL explicitly mentioned the ability to add new computing resources to meet an increased demand from its users. We believe that only the addition of computing resources to a cloud infrastructure is, by itself, not always enough, since part of the resources may become overloaded if the system load is not appropriately distributed. For this reason, our software infrastructure implements load balancing mechanisms for handling mobile devices connections at the gateways and also for data processing at the processing nodes.

In respect to power management, the work Rahmani et al. [24], Cubo et al. [25], Balasubraman and Stranieri [26] and Doukas et al. [29] are based on the use of stationary gateways where the device power consumption is not an issue, since they are connected to an uninterrupted power source. Reddy et al. [28] provides mobile gateways, but does not describe specific power management mechanisms. In M-Hub, the Energy Manager component triggers events that adapt the behavior of services according to the battery level. The use of in-network processing using Complex Event Processing rules or Java code can also preserve battery by avoiding network communications. Finally, the MR-UDP protocol was also designed to minimize energy consumption since it is based on a lightweight protocol.

Considering the comparisons, we argue that the M-Hub/SDDL covers a larger number of challenges related to the development of AAL systems. We offer a scalable software infrastructure, a reliable communication protocol with support for firewall and NAT traversal and changes of IP address over wireless channels, the capacity to handle a wide range of sensor devices used in the monitoring of patients, efficient power management mechanisms, and flexibility to perform computations in a cloud infrastructure or in the mobile device with support for dynamic loading.
and instantiation of code at runtime. M-Hub/SDDL is applicable in all described AAL scenarios, proving an adequate support for several patient’s degree of mobility and degree of physical and cognitive skills for interacting with the monitoring systems.

Beyond this comparison with other research groups initiatives, it is important to explain that the SDDL and M-Hub have been the subject of discussion and evaluation in some of our previously published work in opportunistic mobile sensing [16] [30] and scalable data distribution fields [9], [14] [11]. However, these work were not motivated by Ambient Assisted Living scenarios. Here, we focus on healthcare applications, promoting a broad discussion about how the M-Hub and SDDL can be used to address several challenges related to patient monitoring in several AAL scenarios, including a case study that shows how M-Hub/SDDL was used to implement an AAL system for the human movement activity recognition. In addition, we present and discuss the results of a new SDDL scalability evaluation, using a specific AAL application. Finally, the present work provides a more detailed explanation about the architecture and some functionalities provided by SDDL and M-Hub components, especially regarding how the infrastructure handles concurrent requests sent by mobile clients.

7. CONCLUSIONS AND FUTURE WORK

The development of systems for Ambient Assisted Living (AAL) gained prominence with the increasing concern for the health of elderly patients, especially those with chronic diseases. This paper presents a software infrastructure targeting AAL systems based on IoT and cloud computing composed of two main components: the Mobile Hub (M-Hub) that runs on mobile devices, and the Scalable Data Distribution Layer (SDDL) that provides a cloud-based infrastructure. It provides a deep explanation of how the proposed software addresses several relevant challenges related to the development and implementation of AAL systems, especially the comprehensiveness of AAL scenarios, support for reliable communication and heterogeneous technologies, scalability, and power management.

Among the main features of M-Hub/SDDL we highlight: a) the support for patient mobility in various AAL scenarios; b) the provision of a reliable communication protocol with support for firewall/NAT traversal and changes of IP address over wireless channels; c) the ability to interact with various types of sensors and short range wireless communication technologies; d) the support for dynamic download and execution of code at the mobile device expressed as Complex Event Processing rules or plain Java; e) energy management at the mobile device, adapting the software operation according to the battery status; and f) the ability to add computing resources on demand and to balance the system workload on the cloud side.

This paper provides a deep explanation of how the M-Hub/SDDL could be applied for the development of AAL applications in various scenarios: Nursing Home, Assisted Living, Home Care, and Mobile Health. It also illustrates a case study comprising the development of a Human Movement Activity Recognition System using the provided software infrastructure.

Experimental results demonstrated that the M-Hub is able to timely deal with the automatic and opportunistic sensor discovery and data gathering requirements considering all the described AAL scenarios and that SDDL is able to handle a large number of clients (patients) providing system scalability in respect to connectivity management and context data processing.

We are currently investigating the provision of Quality of Context (QoC) mechanisms to the M-Hub/SDDL considering three dimensions: (i) quality related to the sensors devices; (ii) quality of the provided context data; and (iii) quality of the context data distribution infrastructure.

ACKNOWLEDGEMENT

The authors would like to thank FAPEMA (State of Maranhão Research Funding Agency), CAPES (Brazilian Federal Agency for the Support and Evaluation of Graduate Education) and CNPq (National Council for Scientific and Technological Development) for the research scholarships and partial financial support of this work. The Mobile Hub project is supported by the PUC-Rio Microsoft Open Source Alliance.
REFERENCES


Table VI. Table of acronyms

<table>
<thead>
<tr>
<th>ACRONYM</th>
<th>MEANING</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAL</td>
<td>Ambient Assisted Living</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>AppA</td>
<td>Assistive Patient monitoring cloud Platform for Active healthcare applications</td>
</tr>
<tr>
<td>BLE</td>
<td>Bluetooth Low Energy</td>
</tr>
<tr>
<td>CEP</td>
<td>Complex Event Processing</td>
</tr>
<tr>
<td>CHMS</td>
<td>Cloud Framework for Health Care Monitoring System</td>
</tr>
<tr>
<td>CoT</td>
<td>Connection Time</td>
</tr>
<tr>
<td>CPU</td>
<td>Central Processing Unit</td>
</tr>
<tr>
<td>DCPS</td>
<td>Data Centric Publish-Subscribe</td>
</tr>
<tr>
<td>DDS</td>
<td>Data Distribution Service</td>
</tr>
<tr>
<td>DPS</td>
<td>Data Processing Slice</td>
</tr>
<tr>
<td>DPSLB</td>
<td>Data Processing Slice Load Balancing</td>
</tr>
<tr>
<td>ECG</td>
<td>Electrocardiogram</td>
</tr>
<tr>
<td>EMG</td>
<td>Electromyogram</td>
</tr>
<tr>
<td>EPL</td>
<td>Event Processing Language</td>
</tr>
<tr>
<td>GPRS</td>
<td>General Packet Radio Services</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Position System</td>
</tr>
<tr>
<td>IMedBox</td>
<td>Intelligent Medicine Box</td>
</tr>
<tr>
<td>IoT</td>
<td>Internet of Mobile Things</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>KNN</td>
<td>K-Nearest Neighbors</td>
</tr>
<tr>
<td>LTS</td>
<td>Long Term Support</td>
</tr>
<tr>
<td>MAIRS</td>
<td>Mobile Activity and Intensity Recording System</td>
</tr>
<tr>
<td>MEPA</td>
<td>Mobile Event Processing Agent</td>
</tr>
<tr>
<td>M-Hub</td>
<td>Mobile Hub</td>
</tr>
<tr>
<td>MR-UDP</td>
<td>Mobile Reliable UDP</td>
</tr>
<tr>
<td>NFC</td>
<td>Near Field Communication</td>
</tr>
<tr>
<td>MTD</td>
<td>Temporary Disconnection</td>
</tr>
<tr>
<td>NAT</td>
<td>Network Address Translation</td>
</tr>
<tr>
<td>OMG</td>
<td>Object Management Group</td>
</tr>
<tr>
<td>P2P</td>
<td>Peer to Peer</td>
</tr>
<tr>
<td>PoA</td>
<td>Point of Attachment</td>
</tr>
<tr>
<td>QoC</td>
<td>Quality of Context</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>RAM</td>
<td>Random Access Memory</td>
</tr>
<tr>
<td>RFID</td>
<td>Radio-Frequency Identification</td>
</tr>
<tr>
<td>RTPS</td>
<td>Real-Time Publish-Subscribe</td>
</tr>
<tr>
<td>S2PA</td>
<td>Short-range Sensing, Presence &amp; Actuation API</td>
</tr>
<tr>
<td>SDDL</td>
<td>Scalable Data Distribution Layer</td>
</tr>
<tr>
<td>SDK</td>
<td>Software Development Kit</td>
</tr>
<tr>
<td>SDT</td>
<td>Service Discovery Time</td>
</tr>
<tr>
<td>SQL</td>
<td>Structured Query Language</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>TR</td>
<td>Time to Receive</td>
</tr>
<tr>
<td>TTR</td>
<td>Total Time to Receive</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless Local Area Network</td>
</tr>
<tr>
<td>WPAN</td>
<td>Wireless Personal Area Network</td>
</tr>
<tr>
<td>WWAN</td>
<td>Wireless Wide Area Network</td>
</tr>
</tbody>
</table>