Tackling Wireless Sensor Network Heterogeneity Through Novel Reconfigurable Gateway Approach

Ch. P. Antonopoulos, K. Antonopoulos, Ch. Panagiotou, N. S. Voros
Computer & Informatics Engineering Dept, TEI of Western Greece, Antirio, Greece
(*)Corresponding author email: cantonopoulos@teiwest.gr

Abstract. Without Gateways comprise critical cornerstones for a wide range of Cyber Physical Systems (CPS) and IoT application scenarios. In this context, the main objective of this paper is to present a novel Gateway able to effectively address all heterogeneity aspects. Additionally, a critical objective of the proposed Gateway is offering advanced features facilitating the dynamic, run time service management. Aiming to offer a comprehensive solution all important aspects are presented in detail while a real-world evaluation is provided. The latter, is achieved by integrating the implemented solution with a real CPS infrastructure enabling i) hands on proof of concept concerning heterogeneity support and offered features and ii) realistic performance evaluation.

Keywords: Wireless Sensor Networks , Gateway Design, Cyber Physical Systems, Heterogeneous Technologies, Message Passing Communication, Practical deployment Implementation, Performance Validation and Verification.

1 INTRODUCTION

The main objective of a Cyber Physical System (CPS) is to tightly integrate cyber and physical objects, which entails addressing a wide range of challenging heterogeneities [1]. Towards this objective the Gateway entity is of paramount importance since it comprises both the physical and architectural point where data from the physical domain are aggregated and effectively transported to the cyber domain and vise versa. The importance of an efficient Gateway is even more emphasized by the multiple scales a CPS system can be envisioned, as well as application scenarios ranging from smart houses to autonomous cars, and from advanced smart medical devices to emergency event detection and autonomous reaction [2]. Wireless Sensor Networks (WSNs) and embedded systems comprise the two main cornerstones of an effective and efficient Gateway design with respect to communication, processing and control, considering heterogeneous and diverse systems. Specifically, for a system to be useful and practical, it is critical to offer adequate performance capabilities and high degree of flexibility with respect to remote monitoring and actuation requirement. In that context, messaging [3] represents a highly considered technology, enabling high-speed, asynchronous, highly reliable machine-to-machine or even program-to-program communication. Such communication is possible by exchanging data packets, called messages, to each other. Channels, also known as queues, are logical path-

ways that connect the programs and convey such messages. A channel behaves like a collection or array of messages shared across multiple Gateways and effectively any other component of the end-to-end CPS infrastructure [3].

Turning our attention to the physical domain respective sources of heterogeneity can be identified. On one hand, the diverse modalities and data types, imposed by versatile and flexible solutions, must be effectively handled by the Gateway architecture. This is important since at that point all different modalities and data are effectively homogenized and managed in a harmonized way. On the other hand, nowadays there is a wide range of heterogenous sensor communication technologies, each characterized by specific and distinct advantages and features. Consequently, an efficient CPS Gateway is required to handle data in a homogenous way aiming to offer common degree of robustness, QoS, synchronization and power efficiency [4].

Furthermore, during the last years due to advancements in areas such as embedded system and hardware design, Gateways are increasingly considered, not only for data aggregation and forwarding, but also to store, fuse and process data in real time. This demand opens us a whole new area of heterogeneities regarding processing capabilities, memory capabilities and data base integration requirements [5].

The heterogeneity reflects the necessity for reconfigurable solutions that could support high level of adaptability to domains with increased diversity like IoT and CPS. The gateway particularly, which is the link between the physical world and the cloud needs to bridge a plethora of communication technologies (with diverse requirements and constraints) with the fragmented world of IoT standards and protocols. A reconfigurable architecture for gateways in the IoT ecosystem will be a game changer towards the design and development of cost effective and efficient CPS and services under the IoT application domain.

Driven by the identified necessity for practical and reliable solutions, this paper presents a complete Gateway architecture yielding critical advantages, features and services able to effectively support a wide range of realistic application scenarios. Based on message passing communication technology and utilizing Commercial-Of-The-Shelf (COTS) technologies, the proposed Gateway is implemented offering an abstraction layer that effectively hides all technology heterogeneity and/or peculiarity. All design and implementation aspects of the involved entities are presented in detail in the following sections. Moreover, the overall performance of the proposed solution is demonstrated at system level by integrating the proposed Gateway to a real CPS infrastructure and focusing on two main aspects: (a) performance robustness and efficiency (in terms of being able to meet demanding application requirements), and (b) resource consumption, which is critical for incorporating the proposed architecture in real embedded systems. The high degree of reconfigurability, extendibility and scalability comprise the main directives that drive the design and implementation phases of the gateway.

The rest of the paper is structured as follows: The Section II discusses the main challenges that drove the design and implementation of the proposed Gateway. Section III offers detailed insight analysis of critical design aspects and decisions, while Section IV presents the implementation of the proposed solution. Section V analyzes and presents performance evaluation and validation results pertaining to communication

capabilities and resource expenditure. Finally, Section VI concludes and summarizes the most important points of this paper.

2 KEY CHALLENGES ADDRESSED

Driven by the requirements posed by, on one hand, nowadays and future CPS applications and, on the other hand, by the end users, the proposed Gateway focuses to address a wide range of emerging challenges

Regarding the low power communication perspective of CPSs, although a plethora of different communication technologies (mainly originating from WSN research domain) are available, they offer diverse characteristics exhibiting high degree of incompatibility. In that respect, the proposed Gateway design supports all prominent short range wireless communication technologies such as IEEE 802.15.4 [6], ZigBee [7], Bluetooth [8], BLE [9], while currently, the support of Z-Wave [10] protocol is ongoing. Also, a critical goal of the design is to facilitate the continuous development and integration of new solutions.

Furthermore, the increased functional complexity calls for new processing capabilities in the future CPS platforms. To address the respective challenge, the presented Gateway design highlights the ability to support data acquisition by different modalities and using different communication technologies to be synchronized, homogenized and processed. In this way, sophisticated load balancing, data merging, QoS, prioritizing and many more mechanisms can be supported. This approach also facilitates real-time data processing and event detection which is of paramount importance in demanding applications such medical and industrial deployments.

A CPS platform is usually comprised by multiple, highly scattered nodes able to interact with the real world, thus supporting a wide range of diverse applications. Therefore, controlling, optimizing, adjusting and enhancing a continuously expanding CPS platform by physically accessing each node or/and component is unpractical, cumbersome and in specific deployment practically impossible. Consequently, it is critical for an efficient Gateway to facilitate real-time remote control of all components as well as on-the-fly functional updates and service optimization. In the context of the proposed solution, remote monitoring, management and control is possible through message passing communication technologies via MQTT and MQTT-SN protocols [11]. However, the most important characteristic of the presented Gateway is the capability provided to the end user/service-provider to remotely reconfigure and seamlessly define a new service, deploy it and activate it throughout the network. Technical details that manifest the design and implementation of a reconfigurable gateway will be presented in the following sections. In this way, the operation of the platform can continuously evolve and be optimized with respect to specific requirements or/and conditions.

3 THE PROPOSED GATEWAY DESIGN

Aiming to offer a modular architecture Figure 1 depicts the proposed design.

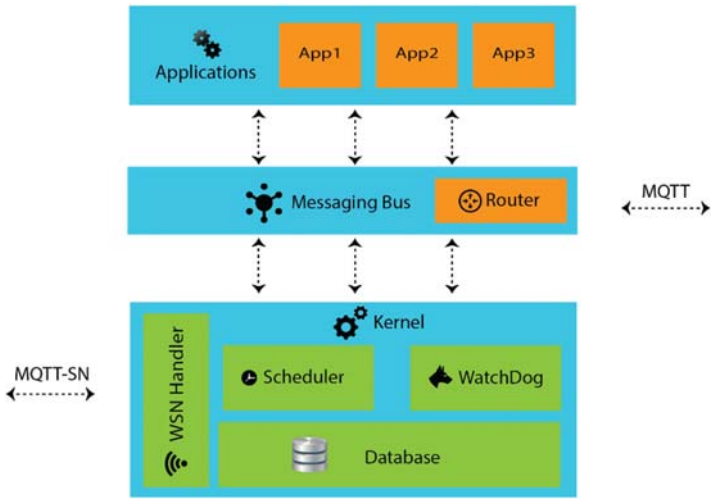

Figure 1. The Proposed Gateway Design

The high-level design, of the Gateway architecture consists from three main components, namely the Kernel, Applications and the Messaging Bus. At the lower layer of the architecture lies the Kernel of the Gateway, where the core modules are deployed. On the contrary, at the top of the architecture, user-defined applications are commissioned, typically requiring a considerably higher degree of flexibility. Finally, in order to assure efficient interconnection between the two aforementioned components, a dedicated intermediate layer, the Messaging Bus, is introduced providing intra Gateway modules communication capabilities and functionalities.

Starting with the internal communication requirements of a modular and reconfigurable design, the primary objective is to use a single communication protocol for all the communications among gateway entities, regardless if they physically reside in the Gateway or not. In that respect, the Message Queue Telemetry Transport (MQTT) [13] protocol utilized for all the communication between the gateway modules comprises a perfectly suiting solution.

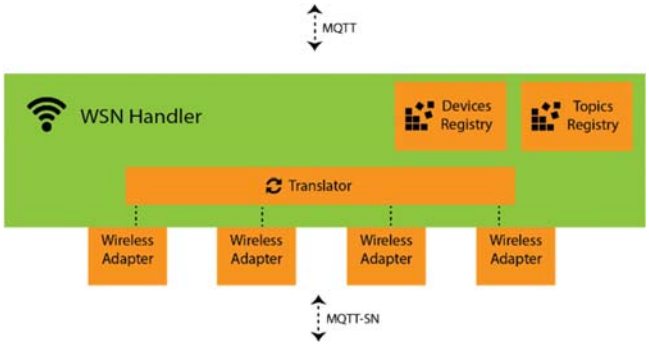

Figure 2. WSN Handler

The proposed design integrates wireless communication adapters in order to offer compatibility with a wide range of prominent wireless sensor networks, like Bluetooth, BLE, ZigBee, WI-FI, Z-wave etc. To effectively tackle the heterogeneity caused by the diverse wireless communication technologies, the MQTT-SN [14] protocol has been exploited. The main goal of the proposed WSN handler components is to handle all the heterogeneity and complexity, of the abovementioned communication technologies. An abstract view of the WSN Handler is depicted in Figure 2.

As shown at the bottom level of the module, specific wireless communication adapters are developed, which forward all the WSN generated traffic to the Translator module. Then, the Translator receives the MQTT-SN messages and transforms them to valid MQTT messages. Finally, converted MQTT messages are forwarded to the connected MQTT Server, which in our proposed design is integrated into the Messaging Bus.

Device and Topics Registry comprise critical modules assuring the adequate operation of the WSN Handler. Specifically, topics registry contains the MQTT topics and the corresponding MQTT-SN topics Identifiers. Devices Registry tackles the challenging heterogeneity caused by the incompatible WSN communication technologies regarding devices addresses.

The Messaging Bus, is the main aggregation point for all incoming and outgoing messages related to the Gateway. Internally, it offers local MQTT connections among the Gateway modules and applications. Externally, it maintains MQTT connections to IoT Back-end infrastructure, for the Gateway to communicate with the outside IP networks. Due to the existence of two different messaging systems, the internal Gateway messaging system (Modules, applications communication), and the external messaging system (Back-end IoT infrastructure communication), the need for a module was raised to bridge these two systems. To tackle this need, the Router plugin is added to the proposed design, inside the Messaging Bus. The plugin performs topic-based routing by intercepting messages from topics, analyzing them and finally forwarding them, to the appropriate, external or internal MQTT topics, without modifying the message content.

The Database module, as anticipated is responsible to provide storage capabilities to the Gateway entity. Every module or application that is connected or executed on the

Gateway, can create its own database tables, and save, fetch or delete data using a generic and simplified API.
The Scheduler module, as the name implies is used for internal gateway tasks, concerning both modules and applications. Mainly there are two main type of tasks:

1. Repeatable tasks, repeatable fired tasks.
2. Future tasks, one time fired future tasks.

The watchdog module's main objective is to monitor all the hardware and software that the Gateway is running. It sends periodically status messages to an IoT Back-end infrastructure, which assist developers to effectively maintain the overall status of the deployed Gateways.
Finally, the Applications module is responsible for the deployment and management of user-defined applications.
Concluding, the proposed design offers a multifaceted and comprehensive architecture able to meet any demand or requirements posed by nowadays and future heterogenous WSN deployments.

4 NOVEL GATEWAY IMPLEMENTATION

The kernel of the Gateway implementation has been written in JAVA programming language, using the Spring [15] framework, thus providing component-based architecture and auto-wiring functionality to the core Gateway modules. As a prominent service-oriented framework the Spring framework is used for the development of the core Gateway. In this way, the implementation yields considerably lower complexity, compared to solutions such as the OSGI [16] framework leveraging the development of loosely coupled components that deliver their functionalities through MQTT topics. In the center of the application component nexus, exists a light and monolithic kernel that exposes the core gateway functionality. Consequently, respective implementation choice facilitates the development of a robust and reliable core, that will not change frequently. On top of this core all the functionalities will be exposed using APIs through messaging topics. Thus, allowing the development and dynamic customization of application components specifically tailored to the needs of the application scenario and in favor of modularity and reconfigurability.
The main implementation approach to structure the proposed Gateway is based on messaging patterns. In that respect, the implemented Gateway consists of modules exposing MQTT APIs. In order for the modules to expose such APIs, they must be connected to the Gateway Messaging Bus, as depicted in Figure 1. The core implementation of the Messaging Bus, is based on ActiveMQ message broker [17]. ActiveMQ is a highly efficient JMS implementation product based on open sources of Apache protocol, with proven maturity and rich features. The default messaging protocol used by ActiveMQ is OpenWire [18]. The developed solution creates a duplex connection, allowing the Gateway to produce and consume messages from the IoT back-end infrastructure. However, such a connection type creates a critical point concerning how to distinguish which messages from the internal messaging system will be forwarded to the external messaging system and vice versa. In that respect, an

ActiveMQ plugin has been developed comprising a topic-based interceptor, depicted as the Router in Figure 1. Its main functionality it to examine MQTT topics names, and performing the bridging in case the corresponding topic bridge has been configured. Additionally, a flexible mechanism is implemented to add/remove bridges to/from the Router plugin. The router waits for messages in MQTT topic router/bridges in order to create or delete a bridge between the two messaging systems. The message payload used for all the MQTT APIs, is JSON [19]. JSON is used because it is a lightweight text-based protocol, with a human-readable format, which is capable to represent complex information, with minimum data.

The database module provides storage capabilities in the Gateway, for both Gateway modules and applications. We choose to utilize the SQLite [20] embedded database since it can be used in a zero-configuration mode and consume very little resources, which make it a valuable choice for an embedded system. Another significant advantage of SQLite is its ability to work in the two following modes, i) In-memory where the data are stored to memory, and ii) In-file, where the data are stored to disk files. In the context of the proposed Gateway, SQLite configured on In-memory mode is used. Again, the access of all the Gateway's modules to the database is accomplished through a MQTT API. In order to allow queries to be executed to the database authors define a simple Query Domain Specific Language (QDSL), using a JSON syntax.

The implemented Applications module is responsible to dynamically deploy and manage user-defined applications that run on the Gateway. Moreover, the APIs exposed by each module are used to monitor the health status of the applications and control their deployment cycle. The exposes a MQTT API that listens for incoming messages correlated with deployed applications, enabling to act on the application, such as start/stop an application.

The module is capable to deploy applications developed in any programming language. Due to this heterogeneity support, in applications, we need a unified way to structure applications, which at the same time will be simple, and allow the developers to structure their applications, in any way fitting their needs. In that respect authors choose to package our applications into compressed files like zip files, with the only requirement to contain a file, namely deploy.json, which will contain all the necessary instructing the module to deploy the application, and the executable file.

Facilitating the described functionality, an online repository has been developed and deployed on the IoT back-end infrastructure. This repository, aims to deliver to the gateway, applications that could modify or enhance its functionality

The Scheduler module is a dynamic scheduling mechanism, exposing functionalities for the modules and applications that are running (or are connected) to the Gateway, enabling them to run repetitive or simple tasks. In order to avoid running subprocesses or multiple thread inside applications or modules, and given the fact that a messaging system can be used for triggering events, a simplified MQTT API is exposed to register future or repetitive tasks, using the following topic, scheduler/tasks.

5 PERFORMANCE EVALUATION

The main goal of this section is to demonstrate and evaluate the performance of the proposed Gateway implementation. Initially, aiming to evaluate time constrained behavior and performance robustness, the round-trip delay of a message sent by the Wireless Sensor Nodes until the respective response is received has been evaluated. Furthermore, in order to argue on the resource conservative design and scalability capabilities, CPU and memory usage are recorded on the Gateway during its full operation. Additionally, considering the efficient addressing of WSN heterogeneity challenge, both Bluetooth and IEEE 802.15.4 devices are utilized in the conducted experiments. The hardware platform hosting the proposed Gateway implementation is a typical Raspberry Pi device featuring 1GByte RAM memory and a 64-bit Quad-core ARMv8 1.2GHz [21]. For that purpose, the proposed Gateway is integrated in a real world Ambient Assisted Living House (AAL HOUSE) [12] controlled and monitored by a complete MQTT based infrastructure. AAL HOUSE is practically a smart home environment that is used for installing and evaluating the latest CPS technologies in real world scenarios.

Experimental Setup

Aiming to offer a useful, objective and multifaceted evaluation of the proposed Gateway design, the following network parameters have been taken into consideration:

1. Traffic data rate of 5 transmitted messages per second.
2. A data message size of 16 bytes, which is typical for WSNs in cyber-physical applications.
3. 1 up to 6 concurrently transmitting nodes.

The first and second parameter fulfill the use case scenario “Elder Daily Activity Monitoring and Support”, presented in the previous section, that requires the generation of 80 bytes per second. The values selected for the specific scenario, as well as the number of concurrently operational nodes, correspond to parameters required for a typical number of persons residing in realistic medium to large home environment. Regarding the measuring of the round-trip delay of a message, the following scenario has been implemented.

Initially, a service is initiated in the back-end infrastructure, that receives the measurement messages and, without performing any manipulation, sends them back to the Gateway. Sensors, before sending the message to the Gateway, they tag the transmission time. When the Gateway receives the message, it forwards it to the Messaging service, which in turn conveys it to the MQTT based network service. As mentioned before, when our service receives the message it sends it back to Message service, which forwards it again to the Gateway. At this point, the Gateway extracts the payload from the message in order to find the destination sensor mote towards which it will send it. Finally, when the sensor mote receives the message, it calculates the delay between the time of the message arrived and the time of message transmitted, that have been tagged when the message was initially transmitted.

The evaluation undertaken is conducted based on Shimmer platform [22], which offers both Bluetooth and 802.15.4, communication interfaces. Shimmer nodes' software stack is based on the open source TinyOS operating system. The Gateway is a standard x86 PC running Linux operating system.

Experimental Results

In order to expose the effect of heterogeneous communication technologies three different case are considered. On one hand, delay is measured considering solely devices transmitting messages using Bluetooth interfaces. On the other hand, delay performance with respect to solely IEEE 802.15.4 based wireless interfaces is considered. Finally, mixed network scenarios are formed, where half of the sensors send messages through Bluetooth and the rest through IEEE 802.15.4 interfaces.

Figure 3 depicts the time interval for a message to be sent from a device and get back the respective response. As shown, when only Bluetooth devices transmit messages, the mean delay ranges from 120 msec up to 150 msec, when 6 devices transmit concurrently.

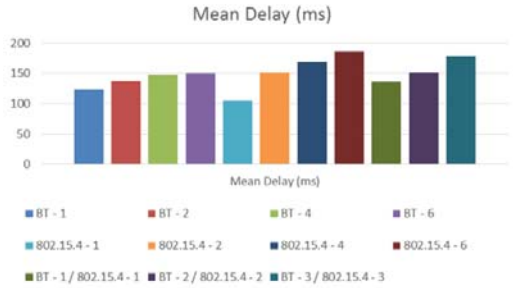

Figure 3. Messages' mean delay in ATLAS

Considering networks comprised by solely 802.15.4 based nodes, it is depicted that when there is not competition (i.e. single transmitting node), IEEE 802.15.4 mean delay is slightly reduced compared to Bluetooth cases. However, when there are concurrently competing nodes (i.e. 802.15.4-2/ 4/ 6), measurements indicate a steadily increasing delay overhead compared to Bluetooth graphs. Indicatively, considering six (6) concurrently transmitting nodes mean delay reaches up to 185msec, corresponding to a 23% delay increased compared to Bluetooth respect case.

Finally, in the mixed networking scenario the results are approximately in the middle compared to cases based specifically on either IEEE 802.15.4 or Bluetooth. Consequently, the implemented Gateway effectively handled these two heterogenous communication technologies with no noticeable overhead. Additionally, it exhibits considerable stability with respect to heterogeneous technologies and varying number of sensors.

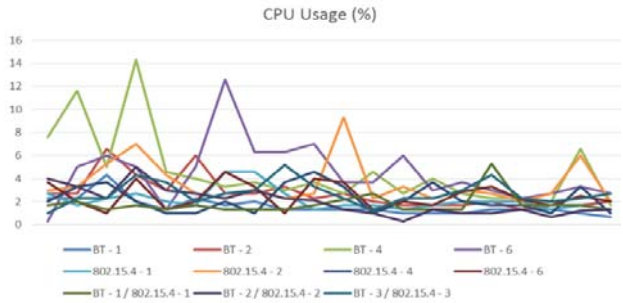

Figure 4. CPU usage in ATLAS WSN devices

Moving on to resource consuming aspect of the evaluation, Figure 4, depicts, the CPU usage of the Gateway, when sensors transmit data. As shown, the usage ranges mainly from 2% up 5% for all scenario, with few spikes, which reach up to 14%. Based on log file analysis, the latter originates from Bluetooth devices transmitting data.
Finally, Figure 5 exhibits the memory usage of the Gateway when sensors transmit data. As shown, the usage ranges from 60 MB on low traffic scenarios, up to 80 MB when high traffic scenarios are imposed. Both memory and CPU utilization measurements appear to be quite low with respect to the demands which can be accommodated by nowadays embedded systems such as Raspberry PI [21] or Intel's Edison [23] based platforms.

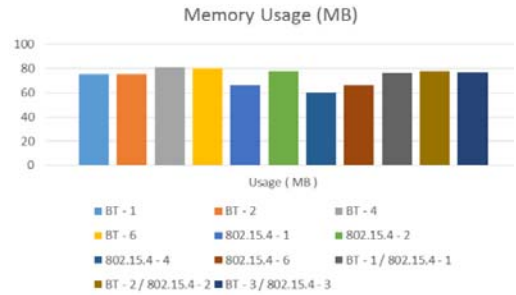

Figure 5. Memory usage in ATLAS WSN devices

Summarizing, from a communication delay point of view it is shown that the number of concurrent transmitters comprises the most important factor affecting performance. Bluetooth is based on connection-oriented and highly robust behavior, exhibiting the smallest delay deviation with respect to competing nodes. However, in all cases, a time constrained behavior is observed, which is less than 200 msec for end-to-end round trip delay and advocates the use of the proposed platform both in conservative as well as demanding smart home environments. From a Gateway processing power point of view, a quite reliable performance was also recorded which can be supported by nowadays single boards computer systems such as Raspberry Pi. This provides an objective proof-of-concept regarding the practical aspects of ATLAS with respect to

heterogeneity support, flexibility and scalability. Finally, the presented performance evaluation results validate the utilization of ATLAS' architecture in the context of commercial solutions.

6 CONCLUSIONS

Over the last few years, Cyber Physical Systems appear as one of the most prominent research areas able to unite the physical and cyber domains in the context of heterogeneous and diverse application scenarios. However, respective solutions to be truly useful, practical and widely adopted there is a critical need for novel Gateway architectures tackling heterogeneity while exhibiting high degree of flexibility and reconfigurability. Moreover, the interconnection of Cloud and CPS in the context of the IoT paradigm evolves, the need for ubiquitous computing rapidly increases. In that sense, data fusion, processing and knowledge extraction should be stratified along the chain of the IoT communication channels.

Therefore, the impact of novel gateway architectures that are the link between the physical world and the cloud is critical since, they are assigned with the key role to provide interconnectivity and maximum application availability. In that context, recent trends in cloud computing optimization and IoT is the distribution of analytics and data processing closer to the data source, also known as edge computing. Driven by this requirement, in this paper we present and analyze a novel, comprehensive, and reconfigurable Gateway architecture based on message passing communication paradigm. All aspects comprising a complete, efficient and versatile novel architecture effectively addressing respective requirements that are presented in detail. Additionally, the proposed architecture has been evaluated in the context of a real world smart home, under the "Elder Daily Activity Monitoring and Support". The reconfigurability of the gateway outshines when scenarios considering different WSN communication technologies and workload patterns are applied. At the same time, the gateway exhibits, in all cases, robust behavior and reliability with respect to time constraints and application requirements.. Furthermore, resource requirements have been measured revealing rather conservative demands which can be easily met by most COTS embedded platforms.

Acknowledgment

This study is part of the collaborative project RADIO which is funded by the European Commission under Horizon 2020 Research and Innovation Programme with Grant Agreement Number 643892.

References

1. Fei Hu, Cyber Physical Systems - Integrated Computing and Engineering Design. CRC Press, 2014

2. Ch. Antonopoulos, et al. "Robots in assisted living environments as an unobtrusive, efficient, reliable and modular solution for independent ageing: The RADIO perspective", 11th ARC 2015, Bochum, Germany, Volume: Springer LNCS 9040, pp. 535 - 546
3. Hophe Gregor, Bobby Woolf, Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions, 2004
4. A. Rajhans et al., "Supporting Heterogeneity in Cyber-Physical Systems Architectures," in IEEE Transactions on Automatic Control, vol. 59, no. 12, pp. 3178-3193, Dec. 2014.
5. W. Kang, S. H. Son and J. A. Stankovic, "Design, Implementation, and Evaluation of a QoS-Aware Real-Time Embedded Database," in IEEE Transactions on Computers, vol. 61, no. 1, pp. 45-59, Jan. 2012.
6. IEEE 802.15.4 Specification, http://standards.ieee.org/about/get/802/802.15.html
7. Zigbee specification, January 2008
8. Bluetooth, Specifications (SIG), 2001, Version 1.1
9. Bluetooth SIG (Hrsg.): Specification of the Bluetooth System: Covered Core Package version:4.0, Juni 2010
10. OpenZwave. (n.d.). Retrieved June 2013, from openzwave Google code site: https://code.google.com/p/openzwave/
11. K. Govindan and A. P. Azad, "End-to-end service assurance in IoT MQTT-SN," 2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC), Las Vegas, NV, 2015, pp. 290-296.
12. http://aalhouse.esda-lab.cied.teiwest.gr/index.php/en/
13. IBM MQTT Protocol Specification. Access Dec 22, 2015, http://public.dhe.ibm.com/software/dw/ webservices/ws-mqtt/mqtt-v3r1.html.
14. MQTT For Sensor Networks (MQTT-SN) Protocol Specification, Version 1.2, http://mqtt.org/new/wp-content/uploads/2009/06/MQTT-SN_spec_v1.2.pdf, Andy Stanford-Clark and Hong Linh Truong, November 2013
15. https://projects.spring.io/spring-boot/
16. Lai C., et al. , "OSGi-based services architecture for Cyber-Physical Home Control Systems", Published in: Journal Computer Communications, Volume 34 Issue 2, February, 2011 Pages 184-191
17. Yin J., et al, Design and Implementation of Intelligent Load-balancing Heterogeneous Data Source Middleware Based on ActiveMQ and XML, 2015 International Conference on Industrial Informatics-Computing Technology, Intelligent Technology, Industrial Information Integration
18. http://activemq.apache.org/openwire.html
19. JSON, http://www.json.org/
20. Owens, Mike. The Definitive Guide to Sqlite[M]. Springer-Verlag New York Inc, 2005.8.
21. Raspberry PI, https://www.raspberrypi.org/
22. Shimmer: http://www.shimmer-research.com/
23. Intel Edison, http://www.intel.com/content/www/us/en/do-it-yourself/edison.htm