Computer Standards & Interfaces 56 (2018) 83–100

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

Middleware and communication technologies for structural health monitoring of critical infrastructures: A survey

Luis Alonso [a], Javier Barbarán [b], Jaime Chen [a,*], Manuel Díaz [a], Luis Llopis [a], Bartolomé Rubio [a]

[a] Department of Languages and Computer Science, University of Málaga, Málaga, Spain
[b] SoftCrits (Software for Critical Systems), Málaga, Spain

ARTICLE INFO

Keywords:
Structural health monitoring
IoT
Wireless Sensor Networks
Critical infrastructure

ABSTRACT

Critical Infrastructure Protection (CIP) has become a priority for every country around the world with the aim of reducing vulnerabilities and improving protection of Critical Infrastructures (CI) against terrorist attacks or natural disasters, among other threats. As part of CIP, Structural Health Monitoring (SHM) is defined as the process of gathering basic information that allows detecting, locating and quantifying vulnerabilities early on (fatigue cracking, degradation of boundary conditions, etc.) thereby improving, the resilience of the CI. Recent advances in electronics, wireless communication and software are expected to open the door to a new era of densely connected devices sharing information worldwide, known as the Internet of Things (IoT), in which Wireless Sensor Networks (WSNs) play an important role. The combined use of IoT/WSNs together with industrial sensors in SHM provide an ad-hoc, inexpensive and easy way of deploying a monitoring system, where data can be shared among different entities. SHM requirements are challenging and diverse and therefore several different technologies may be used in the same deployment. At the same time the use of a middleware can substantially simplify and speed up the development of applications for SHM. Taking into account the challenges of SHM systems, this paper provides a review of the most novel and relevant wireless technologies and a state-of-the-art middleware for WSNs focusing on SHM specific requirements.

1. Introduction

Structural Health Monitoring (SHM) is defined as a process whose objective is to obtain information about the condition and behavior of a structure over time [1]. It is an important technique for monitoring the resilience of Critical Infrastructures (CIs) such as electrical grids, oil and natural gas systems, nuclear power stations or transportation networks. The monitoring process consists of the continuous compilation of the most representative parameters that indicate the state of a structure. The selection of these parameters depends on several factors such as the type of structure, its purpose, the construction materials and environmental conditions. In general terms, these parameters can be mechanical (stress, displacement, deformation), physical (temperature, humidity) or chemical (pH, oxidation of metal). Observation and compilation of parameters can be done both locally (observation of the behavior of a specific material) and globally (observation of the structure as a whole). Typically, innovative structures incorporating new building materials require local monitoring, while conventional structures as well as older constructions require global monitoring. The duration of the monitoring is also established according to the type of structure, which may be short (or temporary) or medium/long (or permanent) term. Continuous monitoring is being used more and more often, even in stages like the construction, which helps engineers to understand the real behavior of the structure while it is being built. It also helps detect construction errors that may affect the future operation of the structure.

A traditional, wired SHM system includes three essential components: a sensor system, an information processing system (including the acquisition, transmission and storage of data) and a health assessment system through the appropriate analysis of information. This traditional system has considerable disadvantages: 1) the high cost, as a result of long data communication cables; 2) low productivity and efficiency, since the deployment of thousands of cables is a labor intensive and time consuming task; 3) low flexibility because of having to deploy extra cables each time new sensors are added to the system. Continuous technological advances in Micro-Electro-Mechanical Systems (MEMS), inte-

* Corresponding author.
E-mail addresses: lalonso@lcc.uma.es (L. Alonso), barbaran@softcrits.es (J. Barbarán), hfc@lcc.uma.es (J. Chen), mdr@lcc.uma.es (M. Díaz), luisll@lcc.uma.es (L. Llopis), tolo@lcc.uma.es (B. Rubio).

https://doi.org/10.1016/j.csi.2017.09.007
Received 24 March 2017; Received in revised form 21 September 2017; Accepted 21 September 2017
Available online 22 September 2017

grated circuits and wireless communication are leading to the production of low-cost platforms that efficiently integrate computing, sensors and wireless communication capabilities. This has had a significant impact on the boom of so-called Wireless Sensor Networks (WSNs) [2]. As a way to overcome the disadvantages of traditional wired SHM systems, WSN-based SHM systems look promising. These systems have many advantages: 1) low cost, by eliminating the wiring of traditional systems; 2) high efficiency, since the installation of wireless sensor nodes is easy; 3) high flexibility, facilitated by the ease of updates, adding, removing and replacing nodes. In addition, the considerable reduction in the size and low cost of MEMS-based sensor nodes and the improvement in their performance make it possible to deploy dense wireless sensor networks, so that the quality of the SHM improves considerably by being able to analyze and correlate data from many strategic points of the CI.

However, while WSNs have been successfully used over the past 15 years in a wide range of applications ranging from environmental monitoring to energy efficient building management, current WSN-based monitoring systems do not meet the stringent requirements of Quality of Service (QoS) for the scope of the SHM of CIs, such as reliability, fault tolerance, synchronization, real-time response and efficiency in energy consumption [3]. In recent years, progress has been made in defining protocols and technologies that provide a certain level of these types of requirements [4]. However, their integration for the reliable development of SHM systems is a highly complex task and difficult to approach if a good methodology is not followed, taking into account the different requirements associated with the type of monitoring to be carried out. The experience of more than a decade has made it clear that a generic monitoring system cannot be built to supervise all CIs, rather it is necessary to carefully analyze each particular case in order to design the most appropriate system.

Relaying on our experience in past projects [5,6], this paper proposes a set of challenges to research and innovate in the context of WSNs for the SHM of CIs. Additionally, we draw attention to two important issues from these challenges, conducting a survey of communication technologies and middleware support. We analyze the communication technology traditionally used in the SHM of CIs and we study new alternatives that look promising in this field. The choice of which communication technology to choose according to the monitoring requirements (permanent or temporary, indoor or outdoor, short or long distance data transmission, continuous or eventual communication) is an important issue when building a specific system. On the other hand, the use of middleware for programming WSNs in general and WSN-based SHM systems in particular, due to the heterogeneity and density of the deployed networks, is of vital importance. A middleware has the ability to remove the programmer from complex aspects of WSN such as the handling of wireless communications (routing protocols, discovery of nodes, etc.), power management, microcontroller low-level programming and synchronization with other devices. A lot of surveys on middleware approaches for WSNs have appeared in the last decade [7–9], but only a few of them have focused on WSN-based SHM systems [10] or IoT-based SHM systems [11] and only some of those have centered on CI [3]. We think both are important issues and must be considered jointly. A deployed WSN-based SHM system can be composed of several communication technologies situated in different points in order to satisfy the multiple requirements established for a CI monitoring system. The use of a middleware that offers a good high-level programming model and abstracts the designer from low level issues, mainly the different wireless communication technologies used, is very important in general and crucial in the particular scope of SHM for CIs.

The rest of the paper is structured as follows. Section 2 outlines different challenges for WSN-based SHM systems. In Section 3 we analyze the different communication technologies and protocols commonly used in WSNs and we present new proposals for their possible use in the context of SHM of CIs. Section 4 reviews some of the existing commercial WSN solutions and research-based WSN deployments for SHM. Section 5 details middleware presented in the literature. Finally, conclusions are presented in Section 6.

2. Challenges of WSN-based SHM of CI

We participated in a Spanish project called Fastrack [5] funded by the Spanish Government's FEDER program. The main objective was the design of a new environmentally and economically sustainable slab track system for high-speed trains (faster than 250 km/h). In the context of this project, we designed a low-cost monitoring platform that is integrated in the slab track, inserted inside it during its construction, so that it can be used both during the installation and in the maintenance phases of the infrastructure [12]. By incorporating this monitoring functionality the slab track becomes an active element, capable of monitoring and reporting information about the environment such as vibration performance, distance and inclination and also assisting operators in the installation phase. Different communication technologies were analyzed and used to get information from the sensor platform [13]. In addition, we have developed a high level middleware called PS-QUASAR [6], based on a simple publish/subscribe model suitable for WSN-based CI protection to facilitate the system's programming.

Taking into account the lessons learned in these projects and in the KAMIC project [14], the project we are currently involved in, we propose the following challenges to improve research in the context of WSN-based SHM of CIs.

2.1. Encapsulated sensor nodes that allow their full integration into the CI

It is very important to design an adapted casing where the sensor nodes are installed, which in turn is inserted into the infrastructure being monitored, for example inside a hole made in the CI structure. The packaging has to be sealed and protected from unauthorized attacks and also from the weather. However, if proper attention is paid to how the casing is designed, this approach will allow operators to easily change the node if necessary (breakage, change of monitoring type, etc.) or remove it in order to change a dead battery. In this way, the monitoring system is integrated into the structure itself, either when it is first built or afterwards, obtaining what we could call an Intelligent Critical Infrastructure (ICI). This constitutes an important innovation with respect to the traditional deployments of SHM systems, which are carried out by placing the different devices with sensor nodes at different points on the structure to be controlled.

2.2. Self-installation of HW/SW components

The engineers in charge of an SHM system of a CI are experts in establishing the appropriate requirements and analyzing the structural health of the structures, but they are usually not knowledgeable about WSN technology. Considering this, ideally engineers should be provided with a kit of self-installing HW/SW components to facilitate their tasks. The kit can be configured per case study depending on their established requirements for the planned monitoring.

2.3. New methodologies implemented as decision support systems

The experience of more than a decade has demonstrated that a generic monitoring system cannot be built to monitor the structural health of all CIs, rather it is necessary to carefully analyze each particular case in order to design the most appropriate system. A methodology should establish the steps to be carried out in the construction of a specific system according to the monitoring requirements (permanent or temporary, indoor or outdoor, short or long distance data transmission, continuous or eventual communication) established by the engineer, the type of data analysis to be performed and the technology available. This methodology should be implemented in a software application that facilitates the work of the engineer responsible for designing the monitoring

system, without knowing the details about the technologies used in the HW/SW components for the final deployment.

2.4. Integration of innovative communication technologies

There are new technological advances in communication devices that have not yet been integrated into the SHM of CIs, such as the recent Bluetooth 4.2 (or upcoming Bluetooth 5.0), the new LTE modules of Category 0 of very low consumption or new technologies especially designed, to take IoT into account, such as NB-IoT, SIGFOX or LoRaWAN. The study and development of HW/SW components based on these new technologies and their integration into the monitoring systems is an interesting line of research. In addition, these new devices will simplify the access of CIs to the IoT so that they can become smart CIs, in line with the Smart City concept, in which a great number of cities around the world are involved.

2.5. Development of appropriate middleware

Due to the heterogeneity and density of the deployed networks in WSN-based SHM systems, the use of middleware to program them is of vital importance. A middleware has the ability to remove the programmer from complex aspects of WSNs such as the handling of wireless communications (routing protocols, discovery of nodes, etc.), power management, microcontroller low-level programming and synchronization with other devices. It provides an easy way to use different communication technologies transparently. Numerous middleware proposals have been put forward for WSNs, but only a few take into account the strict requirements and QoS support necessary in the context of SHM of CIs. Additional effort still needs to be made in this line of research.

2.6. QoS support

The continuous advances in technology have meant that WSNs play a promising role in the field of SHM of CIs [15–17]. Proof of this is that the United States, in its national R&D plan for the protection of critical infrastructures, stated that one of the strategic objectives was "to provide a common national operating framework for critical infrastructures, where core systems would be networks of intelligent sensors". Additionally, the Australian Government, through the Cooperative Security Research Center, have examined and developed solutions to security issues in WSN-based SHM of CIs. In Europe, several projects have addressed this problem [4].

However, current WSN-based monitoring systems do not meet the stringent QoS requirements required for the scope of CI's SHM [3]. When compared to other types of applications, the WSNs used for CI's SHM have some special requirements and characteristics that have to be considered together, such as compatibility between different sensors, their sampling frequencies and modes of operation, bandwidth in data transmission and real-time monitoring, synchronization, reliability, fault tolerance or system life time.

3. Communication technologies

A very important point in an SHM system is the wireless communication module. This determines the efficiency while transmitting the collection of information from each monitoring node to the base station(s), and establishes the limit on the amount of data that the network can handle.

Let us consider that data communication is the most critical aspect regarding energy consumption in a WSN. Therefore, this is the most important aspect to analyze in order to achieve a long network lifespan.

Reliability, the life time of the nodes and the distance of transmission are fundamental aspects to take into account when classifying the suitable technologies. The choice of communication technology will depend on the monitoring requirements and the infrastructure's characteristics.

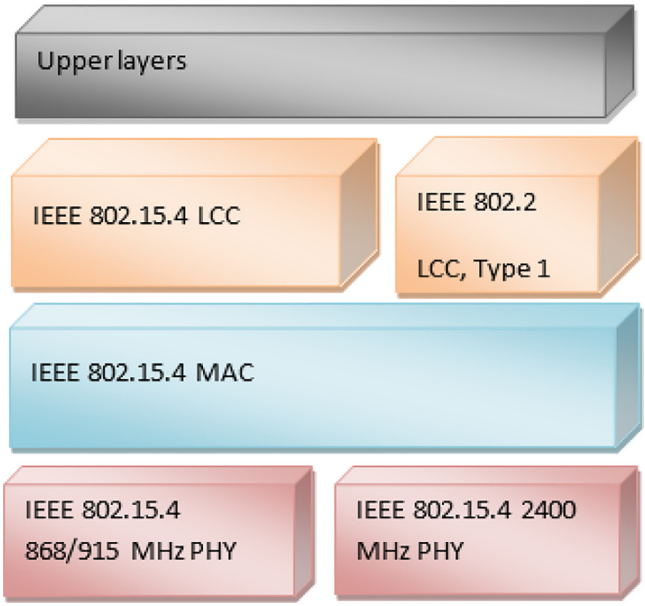

Fig. 1. IEEE 802.15.4 protocol stack.

This section reviews the most widely-used communication technologies (principally using IEEE 802.15.4 at the MAC layer) and some potential candidates to be used in SHM applications.

3.1. Communication based on IEEE 802.15.4

The IEEE 802.15.4 communications protocol [18] is a standard that specifies the physical and data-link levels for small devices with limited communication capabilities. The standard is especially appropriate for scenarios where there are infrequent exchanges of small packets and where energy consumption is a priority. This protocol principally operates in the 2.4 GHz frequency band as well as in the 868 MHz and 915 MHz bands for Europe and North America, respectively. The 2.4 GHz band is divided into 16 channels, each capable of transmitting information at 250 Kbps. The 868 MHz band contains a single channel with a capacity of 20 Kbps while the 915 MHz band contains 10 channels at 40 Kbps each. Fig. 1 shows the different protocols that make up the IEEE 802.15.4 standard.

Access to the medium makes use of CSMA/CA, suitable for networks with little congestion. That is why in highly congested networks, performance declines rapidly due to the waiting time for medium access and collisions. IEEE 802.15.4 has been established as the main physical and link standard for WSNs primarily due to its low power consumption. The rest of the section details the main communication standards for WSNs (network level and higher) all based on this standard.

3.1.1. WirelessHART

The WirelessHART protocol [19] (Wireless Highway Addressable Remote Transducer) is managed by the independent, non-profit HART Communication Foundation. This protocol based on the HART protocol is optimized for wireless communications between control systems and intelligent instrumentation in industrial environments. Control systems are any software application deployed in a heterogeneous set of devices (laptops, SCADAs, mobile devices, etc.). The WirelessHART standard details a wireless communication protocol based on mesh topologies for measurement and control processes in process automation applications. The WirelessHART protocol stack comprises five different levels: physical level, data link level, network level, transport level and application level, as shown in Fig. 2.

The physical layer used is defined in the IEEE 802.15.4 standard. The data link layer however makes use of TDMA with a strict 10ms

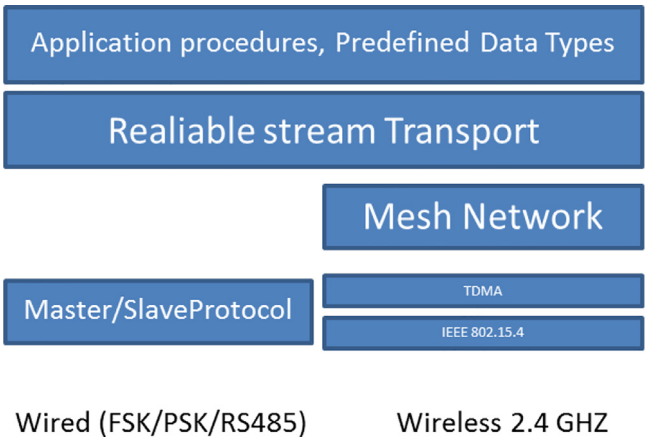

Fig. 2. Hart protocol stack.

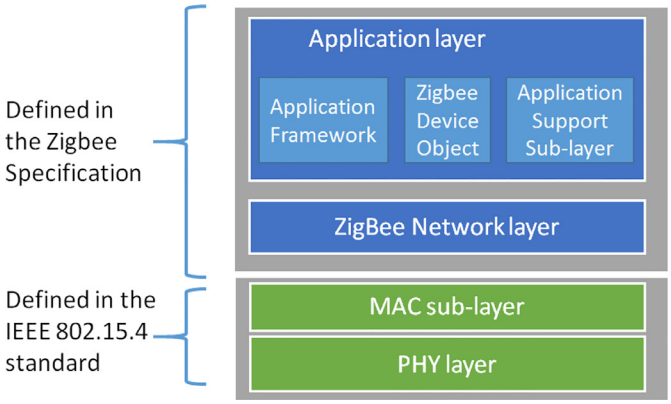

Fig. 3. ZigBee stack.

"time slot" to provide deterministic and interference-free and collision-free real-time communications. The network and transport layers reliably provide communication services based on a high-level command and response scheme. Communications use encryption (AES-128) and integrity check (MIC) systems to provide confidentiality, integrity and availability of information.

This protocol is used in those facilities where security is important. However, we have not found proposals of this nature in the context of SHM for CIs. This could be due to the energy requirements and the monitoring frequency.

3.1.2. ZigBee

The ZigBee standard [20] was developed by the ZigBee Alliance, initially devised as a protocol for home automation applications. Subsequently, in 2007, ZigBee PRO was released and extended the functionality of ZigBee to adapt it to industrial environments. Both versions of the standard make use of IEEE 802.15.4 in the physical and data link layers. ZigBee PRO includes the ability to scan radio channels to determine the least prone to interference, which is then selected and used by all the ZigBee devices in the network. The standard defines a wireless communication protocol designed to be used by small devices grouped in networks of limited size (WPANs) that operate in the bands of 868 MHz, 902–928 MHz and 2.4 GHz. ZigBee allows a maximum communication speed of 250 Kbps between devices with a separation of up to 50 m. ZigBee currently constitutes one of the most widely used protocols in WSNs.

The ZigBee protocol stack is shown in Fig. 3. The physical layer and data link are based on the IEEE 802.15.4 standard. The standard defines two different modes for data transmission: beacon mode and non-beacon mode. Both mechanisms make use of CSMA / CA for medium access. On the other hand, the network layer is responsible for the formation of the network, the allocation of addresses and the routing of messages. This layer participates in the formation of the network through a node discovery mechanism using broadcast messages. AODV (Ad-hoc On-Demand Distance Vector) is used as the default routing protocol for networks organized in a mesh topology. The application layer is composed of a high-level framework for the development and communication of distributed applications. In terms of security, ZigBee incorporates all the security mechanisms proposed by IEEE 802.15.4 such as message encryption, integrity control and access control.

3.1.3. D7AP

The DASH7 Alliance Protocol (D7AP) [21] is an open wireless sensor and actuator network standard, maintained by the DASH7 Alliance. The Alliance published version 1.0 of the specification in 2015. Compared to IEEE 802.15.4 [18] D7AP networks are mainly intended for low-power, medium range and single hop communication. In contrast to IEEE 802.15.4 solutions the aim is not to achieve high data rates, thus it can operate in lower frequencies, use narrow frequency bands as well as more power efficient radio modulations, coding schemes and operation modes that impose lower communication, storage and computational overhead. Simultaneously, the use of lower frequencies allows the effective communication range to be extended at the same energy cost thereby extending the set of possible applications [22]. An additional characteristic is the tree topology deployment to facilitate the management of large networks. In this case, modules must check the channel periodically increasing the power consumption but reduces the latency [23]. This technology may not be suitable for large CIs such as dams where access is difficult and sometimes there is no electricity.

3.1.4. 6LoWPAN

6LoWPAN [24] is a set of standards defined by the "Internet Engineering Task Force (IETF)". These standards allow the efficient use of the IPv6 standard in device networks with limited communication and limited power capabilities. 6LoWPAN initially arose from the idea of using the IP protocol in small devices in such a way as to allow the direct interaction between the Internet and a multitude of different embedded devices. The protocol stack of 6LoWPAN compared to the traditional stack of internet protocols is shown in Fig. 4.

6LoWPAN only supports IPv6 so a layer called "LoWPAN Adaptation Layer" has been included to optimize IPv6 performance over the IEEE 802.15.4 standard. 6LoWPAN normally makes use of UDP packets in the transport layer since it is a lighter protocol than TCP. To control message exchange and error reporting, 6LoWPAN uses the ICMPv6 protocol. Security is provided at the data link level by hop-by-hop encryption and additionally with techniques for checking the integrity of messages and security mechanisms against denial-of-service attacks. Energy efficiency and low cost are well addressed in this technology, however, the growth of this type of network is limited because the management complexity and interference issues can suffer a notable increase with an increase in size of the network. Additionally, its short range means it is necessary to have some additional infrastructure to access the Internet[25]. In this application domain, some test deployments have been done in smart grid systems [26].

3.1.5. ISA100.11.a

The ISA100.11.a - 2009 [27] standard was developed by the ISA100 committee, part of the International Society of Automation (ISA) nonprofit organization. ISA100.11.a provides reliable and secure communications for non-critical control and monitoring applications. The standard defines the protocol stack, system management, gateway specifications, and security mechanisms.

The different layers of the protocol stack according to the OSI model are shown in Fig. 5. At the physical level and part of the data link level, the IEEE 802.15.4 standard is used. The data link layer also includes the ISA100.11a upper data link layer protocol that manages the

Fig. 4. 6LoWPAN stack (left) compared with TCP/IP stack (right).

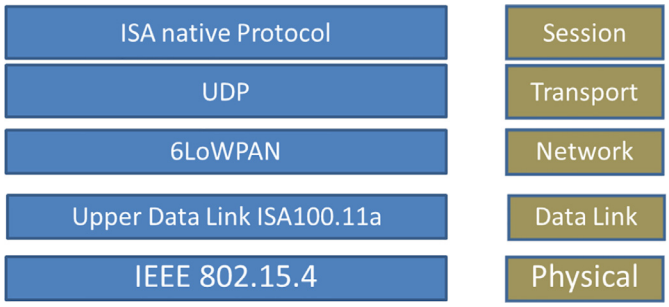

Fig. 5. ISA100.11.a stack.

TDMA mechanism on which the standard operates. The network level provides multi-hop routing capability, QoS requirements management, and other functionalities such as packet fragmentation. In addition, the packet headers are compatible with the 6LoWPAN specification which allows interoperability with the Internet. Security mechanisms are provided at the data link level and at the network level through encryption and authentication techniques.

The network technology predicts a deterministic battery life. It allows CI inspectors to schedule the maintenance work guaranteeing the system's responsiveness. Additionally, the protocol provides interoperability, and a good performance and scalability [28] through diverse techniques such as frequency selection and frequency/channel hopping. Energy scavenging solutions such as solar, heat, vibration and wind could all be suitable power sources. Some examples [29] of industrial applications have used this technology, including machinery health monitoring but there is little mention of it being applied to SHM systems.

3.2. Wireless communications based on pre-existing infrastructures

This type of wireless communication protocol is based on the existence of a preexisting support network deployed on a large scale and covering the individual modules deployed in the local structural health monitoring system.

3.2.1. SIGFOX

SIGFOX [30] is a French company that provides low-power wireless communication systems especially designed for IoT. The term SIGFOX also refers to the devices and communication protocol that the company markets. The protocol relies on the existence of a cellular support network that provides coverage for small communication modules based on Ultra-Narrow Band (UNB) technology on the 868 MHz (Europe) frequency. This technology makes use of small, free sections of frequency bands such as the ISM band and unlike other technologies focuses on allowing a lot of devices to send a small amount of information (about 12 bytes per message). Currently, this technology has good coverage in many European countries such as Spain, France and Ireland.

Finally, these communication modules also allow point-to-point communication without having to make use of the support network. Table 1 shows the main features of SIGFOX.

Table 1
SIGFOX features.

Feature	Value
Frequency	ISM 868 MHz (Europe) ISM 902 MHz (EEUU)ISM 90
Modulation	Ultra-narrow band
Power consumption	TX: 51 mA RX: 16 mA
TX power	24 dBm
Messages	140 messages of 12 bytes per day
Price of license	From 1$ per year and device

Robustness and sensitivity can be increased by using slower transmissions. However, in the context of CIs additional QoS features such as real-time might be necessary. The duty cycle of this protocol makes it less suitable when real-time requirements or frequent sampling are needed. It is necessary to study the timing requirements of the infrastructure to decide on a SIGFOX-based deployment [31]. Vibration sensing, for example, when cars cross a bridge or pass through a tunnel, needs a high data sensing frequency and therefore on-line communication using SIGFOX is not possible. In this case, additional storage devices may be necessary. Throughput is another QoS issue which should be looked at more closely when a large amount of data is exchanged. Furthermore, communication in SIGFOX is unidirectional from the sensor to the cloud and therefore cannot be used in scenarios where control and actuators are necessary.

3.2.2. LTE-M

LTE (Long Term Evolution) [32] is a wireless communications standard, first proposed in 2004, designed to replace existing 4G mobile networks. LTE is incompatible with the current 3G and 2G networks and therefore requires existing infrastructures to be updated. This standard increases the capacity and speed of the current networks through the use of digital signal processing techniques that allows speeds of up to 300 Mbits of download and 75 of upload with a low latency rate while allowing a better mobility management. LTE supports both FDD (duplex division frequency) and TDD (duplex division time). LTE-M is the general evolution of LTE especially oriented to the IoT.

Issues such as scalability, QoS, heterogeneity and battery life are well addressed. Additionally, LTE-M can leverage existing LTE infrastructure. This allows designers to simplify platforms, service management and network architecture. Currently, it is difficult to find LTE-M modules and to integrate them into commercial communication platforms. We think that an interesting proposal would be to test this technology for the SHM of CIs because, in addition to the advantages of this technology discussed above, the interoperability it provides with current devices such as mobile phones is simpler, allowing an increased control over the infrastructure.

3.2.3. Snow

Snow [33] is a network architecture that makes use of TV white spaces (allocated but unused TV channels) to operate. This makes a long communication range possible because VHF and UHF have excellent propagation characteristics over long distances and also a nice obstacle penetration, making communications in urban scenarios more reliable.

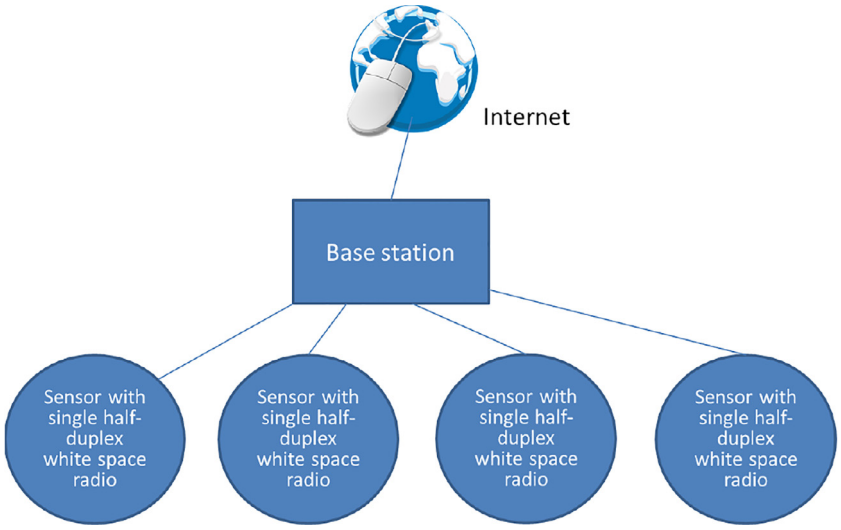

Fig. 6. Snow system architecture.

Table 2
Weightless standards comparison.

	Weightless-W	Weightless-N	Weightless-P
Spectrum	Tv white space	License-exempt ISM	License-exempt sub-GHz ISM/SRD
Bands	470 Mhz–790 MHz	868 Mhz and 915 Mhz	169/433/470/780/ /868/915/923 Mhz
Data rate	1 Kbps–10 Mbps	Up to 500 bps (only uplink)	Adaptative from 200 bps to 100 kbps
Range	5 Km	10 Km	2 Km
Battery life	3–8 years	10 years	3–5 years

This technology offers the possibility of deploying thousands of sensors and connecting them to the base station without the need to use either a multi-hop network or gateways. The scalability and power efficiency is achieved by splitting channels into narrowband orthogonal subcarriers and enabling parallel packet reception in different subcarriers.

As Fig. 6 shows, the architecture is a wireless network based on multiple sensors communicating with only one base station, which is line connected and can communicate via the Internet with any database or back-end system. The sensor nodes are very simple, leaving most of the complexity to the base station, which determines white spaces by accessing a database through the Internet, assuming the location of the sensors is available.

In the context of SHM for CIs, it could be an interesting technology because of its easy deployment, not even needing more than the sensor node itself to connect with the base station. In addition, Snow can reach significant distance communications, over 1.5km, transmitting with relatively low power , which enables a long life, battery-powered sensor node to be used. However, Snow is still in development and modules are not yet available.

3.2.4. Weightless

Weightless [34] is a wireless technology developed by SIG, which has been specifically designed for IoT. It can operate in multiple frequencies, using ultra narrow-band technology and offering very low power consumption, in addition to claiming to have a range of several kilometers. Like Snow, it can operate (only Weightless-W) in the TV white-space spectrum, enabling access to a large amount of the unlicensed spectrum compared to other unlicensed based systems.

Weightless is divided into three different standards applied to different use cases, shown in Table 2. Regarding SHM for CIs, Weightless-N is interesting due to its low cost, range and battery life, even though the communication is only one way.

3.2.5. NB-IOT

NarrowBand IoT (NB-IoT) is a Low Power Wide Area Network (LPWAN) based on narrowband radio technology designed for the IoT. NB-IoT focuses on indoor coverage, low cost, long battery life, and ultra-low device complexity. The NB-IoT technology can either be deployed independently, in unused 200 khz bands or on an LTE base station [35–37]. It can also coexist with 3G and 4G, and benefit from all security, identity, confidentiality, data integrity, and mobile equipment identification that currently exists in mobile networks.

In the context of SHM for CIs, deploying NB-IOT can be difficult since it is not a part of LTE and therefore may need different software or to be deployed in a deprecated GSM spectrum. NB-IOT modules are not yet available. The broadband can reach over a megabit per second so QoS as a real-time requirement can be easily met. In order to facilitate the interoperability of this technology with the other proposals, new middleware must be proposed.

3.3. Other communication technologies

3.3.1. NWave

NWave [38] is a proprietary wireless communication protocol that makes use of UNB technology and has been especially designed for IoT. NWave uses free frequencies within the ISM band to achieve a communication distance of up to 10 km in urban environments. For this it is necessary to have a support network composed of modems and base stations. Table 3 shows the main characteristics of this protocol. NWave does not provide a high data rate and therefore is not suitable for SHM scenarios where a large amount of sensor data is generated and transmitted to the cloud.

3.3.2. LoRaWAN

LoRaWAN [39] is an open standard especially oriented to IoT. It therefore allows communication between a large number of devices making use of the bands below the GHz to obtain a range of Kms. LoRaWAN is attracting the attention of the community due to its low cost

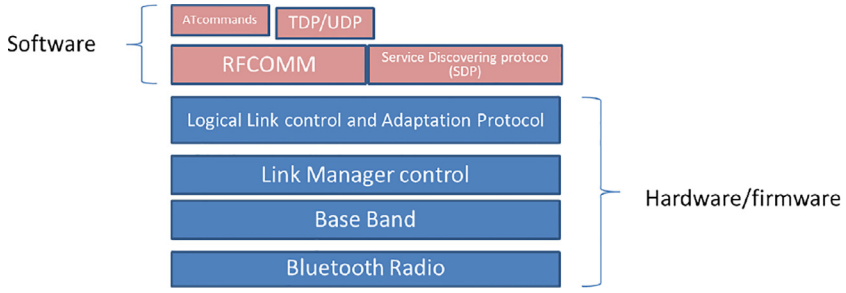

Fig. 7. Bluetooth protocol stack.

Table 3
Nwave features.

Feature	Value
Frequency	<1GHz
Modulation	Ultra-narrow band
Data rate	100 bps
Topology	Star

Table 4
LoRAWAN characteristics.

Feature	Value
Frequency	ISM bands: 109/433/866/915 MHz
Modulation	Ultra narrow band
Data rate	0.3–50Kbps
Power consumption	Very low (over 10 years with two AAA batteries).

and ease of use. However, a number of features are unspecified in LoRAWAN such as roaming, retrying and QoS. As with SIGFOX, for example, using LoRAWAN requires a subscription. Currently the company Semtech is the only producer of communication chips. Table 4 summarizes the main features of this standard. The long-range and low-power nature of LoRa makes it an interesting candidate for smart sensing technology in civil infrastructures (such as health monitoring, smart metering and environment monitoring), as well as in industrial applications [40]. Just like SIGFOX, real-time requirements and scalability must be carefully analyzed for the CI.

3.3.3. Bluetooth

Bluetooth [41] is a short distance wireless communication standard first proposed in 1994 by Ericsson. Initially it was conceived as a replacement protocol for conventional serial communications. Currently Bluetooth is managed by the Bluetooth Special Interest Group consortium made up of thousands of companies in the area of telecommunications and information technology. The standard has been constantly evolving from its version 1.0 to the current version, up to the still under development version 5. This improvements have allowed it to increasingly focus on the field of IoT and improve some of the deficiencies that it had in this sense such as its high energy consumption compared to other protocols in the same field. Bluetooth consists of a protocol stack, shown in Fig. 7, part of which is mandatory and part optional. In addition, it has a security layer that allows you to encrypt communications and provide authentication. Table 5 shows a comparison between the different versions of Bluetooth.

With BLE 4.2 Bluetooth Smart sensors can transmit data over the Internet, simplifying the network architecture to monitor CIs. It includes new functionalities such as geolocation which can reduce the effort of CI node deployments. Issues like security are addressed by providing several features for encryption, trust, data integrity and privacy of the user's data.

Table 5
Bluetooth versions and features.

Version	Range	Data rate	Others	Year
2.0 + EDR	10 m	2.1 Mbps	EDR: Enhanced Data Rate	2004
2.1 + EDR	10 m	3 Mbps	EDR: Enhanced Data Rate	2007
3.0 + HS	10 m	24 Mbps	HS: High Speed. Provides higher data rate by using Wi-Fi physical layer	2009
4.0 + LE	60 m	24 Mbps	LE: Low Energy	2010
5.0	up to 300 m	2 Mbps	Interference detection and prevention	2016–2017

3.3.4. Ingenu RPMA

INGENU is a proprietary LPWA technology, which, unlike most other technologies, operates in the 2.4 GHz ISM band and leverages more relaxed regulations on the spectrum use across different regions [42,43]. INGENU uses a patented physical access scheme called Random Phase Multiple Access (RPMA) [44] Direct Sequence Spread Spectrum, which it employs for uplink communication only. INGENU is leading the effort to standardize the physical layer specifications under the IEEE 802.15.4k standard. RPMA technology is made to be compliant with the IEEE 802.15.4k specification. In comparison with SIGFOX and LoRA, RPMA provides better scalability but the 2.4 GHz band is quite busy and can produce interference. In the case of CIs with real-time requirements additional tests should be carried out.

3.4. Comparison of wireless communications

To help illustrate a comparison between the aforementioned wireless technologies, Table 7 shows the most important features of each technology. The most interesting contribution that has been made in this classification is that we have considered whether or not the communication technology fits in one or more of the four scenarios in which we have classified the SHM of CIs, that is, a continuous or eventual communication and short or long distance data transmission. Table 6 shows the description of each scenario and examples of CI situations that fit into them. Note that technologies marked with a "?" symbol in some scenarios means they are still in development and there is not enough information to confirm or refute its use in that specific scenario.

4. Commercial solutions and real deployments for SHM

4.1. Commercial WSN solutions for SHM

As discussed in the introduction, SHM is a field that has been extensively studied and continues to grow, motivated in part by the rapid development of electronics and wireless communications, in general and

Table 6
Scenario classification.

	Easy access	Remote access
Eventual	Possibility to retrieve data from the site directly. Data is provided when an event occurs or on demand.	d It discards the possibility of being located near where the nodes are deployed. Data is provided when an event occurs or on demand.
	Example.: A tunnel with good accessibility and a monitoring triggered by readings in abnormal range.	Example: Viaduct with non-critical monitoring and monitoring triggered by abnormal readings.
Continuous	Possibility to retrieve data from the site directly. The data will be provided periodically.	It discards the possibility of being located near where the nodes are deployed. The data will be provided periodically.
	Example: A tunnel or small viaduct prone to adverse weather conditions. Continuous monitoring is required.	Example: A large dam in which different parameters are periodically measured to detect any problem as soon as possible.

the IoT, in particular. Proof of this is that there are now a multitude of companies offering commercial systems for infrastructure monitoring, some of which are listed in this section.

RESENSYS [45] is a company that markets systems for the wireless monitoring of structures composed of nodes with a high autonomy (minimum system lifespan of 10 years). They provide a wide range of sensors to monitor a multitude of parameters. They also offer intermediate nodes that collect the information from the final nodes (up to 1000 of them) and communicate with the Internet through GRPS and other communication technologies.

LORD Microstrain [46] offers a complete wireless monitoring system based on IEEE802.15.4 that allows communication distances of up to 20 km with reduced consumption in the range of 10–60 mA. The system also has a wide range of sensors that integrate seamlessly with the communication nodes. There are also gateway nodes that concentrate information from the nodes and direct it to the Internet. In addition, the manufacturer offers an application called Sensor Cloud [47] designed to store and view sensor data in the cloud, which is accessible through an open API. The company MONNIT [48] offers small-sized wireless sensors at a reduced price that communicate in the bands of 868, 900 or 433 MHz. In addition, it has information nodes that use 3G mobile technology to take the information to the Internet. Finally, the system has a software platform for the monitoring and notification of alarms.

BeanAir [49] offers a wireless monitoring system with a multitude of sensors and small nodes that make use of the 2.4 GHz band and the IEEE802.14.4 standard and with an effective communication distance of up to 650 m (LOS). There are additional intermediate nodes that allow communication through ModBus.

An industrial infrastructure monitoring system is offered by SENSEOR [50]. The manufacturer offers a wide range of devices intended for SHM. They also offer sensors built into sensor tags. In this way it is possible to obtain information from fully passive RFID sensors at a maximum distance of 5 m.

Libelium [51] is a marketing company of wireless monitoring devices. They offer complete solutions to the monitoring of infrastructures at the hardware level and it has a wide range of sensors that transparently integrate with the nodes through libraries. The nodes they offer are based on the Arduino prototyping platform.

4.2. Academic WSN deployments for SHM

In this sense, developing WSNs that take into account all the requirements needed for the SHM of CIs represents an important technological challenge. The first proposals of this type of system date back to 1998 [52] and there are many real examples of deployments such as [53–56]. These approaches have carried out real deployments in the metro of Prague/London, in the Jindo bridge (which connects Jindo Island and the southwestern tip of the Korean Peninsula), or in the Basilica S. Maria of Collemaggio (L'Aquila, Italy). In [57] there is an interesting summary of deployments in CI.

Looking at the different WSN platforms taken as a whole, we can see that there has been a clear tendency to use communication protocols based on the IEEE 802.15.4 standard. This is mainly due to its limited consumption. Among these protocols ZigBee is one of the most used as it incorporates additional functionality over the IEEE 802.15.4 protocol such as security, and application management. Overall, we see a general tendency to evolve towards IoT, where the possibility of communication with a multitude of devices is more important than obtaining a high communication speed.

5. Middleware for SHM of CIs

Typically, WSN systems are composed of a series of low-power hardware devices, equipped with a communication module and connected to a series of sensors. The design of these types of systems in most cases is powered by batteries, so a good design of both firmware and hardware is essential to maximize the life of the device. However, today there is a great variety of hardware, both computing and communication for these types of systems, which makes it very difficult for programmers to know all the programming languages and the particularities of each hardware platform. The middleware or frameworks generically are software components whose objective it is to facilitate programming by providing a clearer and simpler interface to access the underlying resources. In the particular case of WSNs, middleware is very useful since, as discussed above, it has the ability to remove the programmer from complex aspects of WSNs such as the handling of wireless communications (routing protocols, node discovery), power management, microcontroller low-level programming and synchronization with other devices. For this reason, in the last decade numerous middleware have been developed that offer a high-level programming model for the programmer (services, publish/subscribe, ...). The use of these software layers allows working with a higher level of abstraction and therefore programming low-level hardware devices with simple primitives available on the Internet, which have emerged as a solution for the IoT.

The large number of middleware for WSNs can be classified according to different criteria. Several approaches like [58,59] advise classifying the existing work by the type of design. In these papers they propose the following classification: A) Event-based B) Service Oriented C) Based on virtual machines D) Agent-based E) Tuple spaces F) Database-oriented G) Application specific. In addition to this classification, it should be noted that there are other open-source middleware created by companies, which are available on the Internet, which have emerged as a solution for the IoT.

SHM requirements need to be studied in detail as we use them as the basis for establishing a set of requirements that middleware should meet. As SHM has different characteristics (permanent or temporary, indoor or outdoor, short or long distance data transmission, continuous or eventual communication), it is not an easy task to provide a generic set of requirements for this type of middleware. Nevertheless, we present a set of middleware requirements based on an analysis of SHM deployments and middleware desirable features:

- Heterogeneity: In terms of the network, as different types of devices and communications might be deployed, so the middleware should be used in the same way in all devices, hiding the details on network heterogeneity from the users. However, data heterogeneity is also important as different magnitudes are going to be measured.
- Abstraction level: The more abstract the middleware is the easier it is for the programmer to work. Providing abstraction means that the

Table 7
Wireless technology comparison.

	Max range	Data rate	Frequency	Channel width	Topology	Cellular coverage	Available modules	Devices per access point	Eventual easy access scenario	Continuous easy access scenario	Eventual remote access scenario	Continuous remote access scenario
WirelessHART	200 m	250 Kbps	2.4 GHz	Channel-hopping	Star Mesh	No	Yes	n/a	No	No	Yes	Yes
ZigBee	200 m	250 Kbps	2.4 GHz/ 868 Mhz	2 Mhz/600 KHz	Star Tree Mesh	No	Yes	n/a	Yes	Yes	Yes	Yes
D7AP	200 m	27.8 Kbps	434 MHz	25 or 200 Khz	Star Tree Mesh	No	Yes	n/a	No	No	Yes	Yes
6LoWPAN	200 m	250 Kbps	2.4 GHz/ 868 Mhz	2 Mhz/600 KHz	Star Tree Mesh	No	Yes	n/a	No	No	Yes	Yes
ISA100.11.a	200 m	250 Kbps	2.4 GHz	Channel-hopping	Star Tree Mesh	No	Yes	n/a	No	No	Yes	Yes
SIGFOX	50 Km	100 bps	868/915 MHz	200 Hz(UNB)	Star	Yes (some countries)	Yes	1M	No	No	No	Yes
LTE-M	<11 Km	<1 Mbps	700/800/ 900 MHz	1.4 Mhz	Star	Yes (in deployment)	No	20k+	No	No	Yes	Yes
NWave	10-30 Km	100 bps	<1 GHz ISM	200 Hz(UNB)	Star	No	Yes	1M	Yes	Yes	Yes	Yes
LoRaWAN	2-15 Km	300 bps–50 Kbps	433/868/780 /915 MHz ISM	125 KHz	Star on star	Yes (some countries)	Yes	1M	Yes	Yes	Yes	Yes
Snow	100 m–1.5 Km	50 Kbps	400–800 Mhz	400 KHz	Single-hop	No	No	Unlimited	?	?	?	?
Weightless-W	5 Km	1 Kbps to 10 Mbps	400–800 Mhz	5 MHz	Star	No	Only for members	Unlimited	?	?	?	?
Weightless-N	3 Km	100 bps	<1 Ghz ISM	200 Hz(UNB)	Star	No	Only for members	Unlimited	?	?	?	?
Weightless-P	2 Km	200 bps to 100 Kbps	<1 Ghz ISM	12.5 KHz	Star	No	Only for members	Unlimited	?	?	?	?
Ingenu RPMA	5 Km	624 Kbps per sector	2.4 GHz	1 MHz	Star Tree	Yes (some cities of USA)	Yes (AP only for rental)	Up to 384.000 per sector.	Yes	Yes	Yes	Yes
Bluetooth	Table 5	Table 5	2.4 GHz	1 Mhz	Master-slave piconet	No	Yes	7	Yes	Yes	No	No
NB-IOT	<15 Km	200 Kbps	700/800/ 900 MHz	180–200 Khz	Point-to-point	Yes (in deployment)	Pre-Order	50k+	?	?	Yes	Yes

middleware hides details from the programmers by providing high level APIs. However, the drawback to this technique is that programmers cannot achieve the desired network efficiency. The abstraction level should not be so high as to impede programmers from configuring the network depending on the SHM application.
- Energy aware: Some sensors might be located at positions along the CI where electricity is not available, therefore the middleware should optimize energy by managing communications efficiently.
- Scalability: In terms of the number of nodes, and also in terms of the number of users and services. Modifying the number of nodes or services should be transparent to the middleware and for the stability of SHM applications using the middleware.
- Security: SHM data can be considered as sensitive data. Therefore, the middleware has to protect the information being sent over the network. On the other hand, access to SHM applications using the middleware has to be authorized.
- Quality of Service (QoS): Depending on the SHM application, QoS requirements have to be adapted, so the middleware should have the ability to adapt to different levels of QoS.
  - Real-time: In terms of packet delivery deadlines it is admissible for SHM applications to miss infrequent deadlines, as long as they are finally delivered. As actions that an SHM application may require will be accomplished anyway, we need soft real-time. However, the user should have the possibility of setting up packet delivery deadlines.
  - Reliability: In SHM applications robustness is crucial as abnormal termination and software failure probabilities should be reduced to a minimum. Middleware should handle the errors and provide backup for the smooth sensing operations in case of failure or system crashes.
  - Fault-tolerance: The middleware should be fault tolerant which means that if any sensor or group of sensors dies the network should overcome the faults.

Middleware for SHM should be able to transparently provide all these features for programmers so the programmer is only aware of the middleware configuration. In the following paragraphs we present an evaluation of the most well-known middleware categories in accordance with the aforementioned criteria. The most representative middleware are analyzed with the aim of determining whether they are suitable for SHM requirements or not. The study of each middleware category presented in this paper is based on middleware theoretical features. In order to determine whether a middleware is suitable for SHM applications we have defined a set of middleware requirements shown above.

The features provided by each middleware in the context of IoT and SHM are summarized in Table 8.

First, some open-source frameworks for WSN/IoT are described in Section 5.1. Then, the most representative middleware in the research literature are surveyed in Section 5.2.

5.1. Open source frameworks for WSN/IoT

5.1.1. Kura

Available at [60] and developed by the Eclipse consortium. It is an open-source framework, Machine-To-Machine (M2M) for the IoT that is based on the installation of the framework in low-cost gateways (raspberry, beaglebone or other industrial options). Kura offers a Java-based application container that complies with OSGI, offering a series of APIs that allow any IoT application to be developed. Kura runs on the Java virtual machine (JVM) and complies with the OSGi standard, a dynamic component service for Java, which simplifies the process of developing reusable code blocks. The APIs offered by Kura provide easy access to device hardware running on serial ports, GPS, watchdog, USB, GPIOs, I2C, etc. The OSGi standard simplifies the management of network configurations, communication with IoT servers, and remote management of gateways.

Kura components are designed as OSGi declarative services so they offer an API and elevate events. Kura components are run in pure Java, however others are invoked through JNI and depend on Linux. On the other hand, IoT world-wide communication standards, such as MQTT [61] (a communications protocol based on the publish-subscribe model), are in general use, and could be used in the case of a WSN [62].

Our study shows that Kura accomplishes hardware heterogeneity, security, abstraction and scalability. However this middleware is designed to be installed on gateways, therefore it is not suitable for resource-constrained devices. All this means that this middleware might be useful for gateways only.

5.1.2. AllJoyn

Available at [63] and developed by the Linux Foundation, it is an open-source framework that helps developers design applications for discovery and communication with others without the need for a cloud. The framework is extremely flexible with multiple features that help in the creation of IoT. Thus, this framework abstracts the complexity of the tasks of the discovery of nearby nodes and applications, and the creation of sessions and secure communications between them. It abstracts transport details by providing an easy-to-use API. Applications communicate with routers bidirectionally. Applications only communicate with other applications through routers.

In addition AllJoyn applications comprise three components:

- AllJoyn App Code, contains application logic. It can be programmed using the Service Framework Libraries or Core Library.
- AllJoyn Service Frameworks Libraries, implement a number of common services, such as notification and control panel. Thanks to the use of this component, the applications and devices can interact with the rest.
- AllJoyn Core library provides low-level APIs to interact with the AllJoyn network (discovery, session creation, ...) It is a framework designed to run on any type of device, including embedded devices running on RTOS.

Alljoyn can run over hard real-time operating systems such as RTOS and therefore seems to be suitable for SHM.

Unlike Kura, AllJoyn middleware provides a version for constrained resources, and also provides different APIs for developers. It provides a proportioned level between abstraction and low-level control so developers get some level of abstraction and also low-level control.

5.1.3. Macchina.io

Available at [64] and developed by Günter Obiltschnig, this is an open-source framework for the rapid development of embedded applications for IoT, which can run on devices running Linux, such as raspberry pi, beaglebone, RED brick or Galileo / Edison. Macchina.io combines the power of Javascript for fast application development with the power and performance of native C ++ code. Macchina.io is based on the POCO C ++ libraries and the JavaScript V8 engine. Macchina.io has been designed in a very modular and extensible way. Fig. 8 shows the architecture of Macchina.io.

The architecture of this framework consists of POCO C ++ libraries, the framework “Remoting”, the Open Service Platform (OSP) and the JavaScript environment. Although the platform has been optimized for use in systems with embedded Linux, it can also be used in other types of applications. The IoT component is at the heart of the framework. Several Open Service Platform and services implement features such as interfaces for devices and sensors, network protocols such as MQTT or COAP, interfaces for Cloud services (Twitter or SMS) and the web interface of the framework itself. There is a PRO version for use by companies that provides additional scalability and security features

Macchina.io is a very module and extensible middleware, which implementings a publish/subscribe messaging protocol. However, it has been designed to be used on non resource-constrained devices, like gateways (as with Kura), so it is not suitable for constrained-resources.

Table 8
Summary of features provided by the middleware.

Middleware	QoS (Reliability)	QoS (Fault tolerance)	QoS (Real-time)	QoS (Energy efficiency)	QoS (Scalability)	Heterogeneity	Abstraction level	Used in WSN	Used in SHM	Security	Support IoT	Programming model	Features
Kura	x (dependant on the underlying protocol)	x (dependant on the underlying protocol)		Dependant on the language and platform used	Good	x (Java)	High	x	x	x	x	Service oriented Publish subscribe (MQTT)	Framework for IoT gateways. It runs over Java virtual machine
Alljoyn	x (dependant on the underlying protocol)	x (dependant on the underlying protocol)	x (via RTOS)	Dependant on the language and platform used	Good	x (Java)	High	x		x		Service oriented	Multiplatform, multilanguage
Macchina.io	x (dependant on the underlying protocol)	x		Dependant on the language and platform used	Good	x (Linux)	High	x		x	x	Service oriented Publish subscribe (MQTT)	Multiprotocol with C++ and Javascript
Hermes	x (IP based)	x		n/a	Good	x (Java)	High					Publish subscribe	Fault tolerant routing algorithm
Green	x (IP based)	x		n/a	Good	n/a	High	x				Publish subscribe	Generic, reconfigurable and reflective middleware
Runes	n/a	n/a		Good	Good	x	High	x				Publish subscribe	Component based middleware
TinyDDS	x	x		n/a	Good	x	High	x		x	x	Publish subscribe	Implementation of the OMG DDS specification, QoS aware
Flexible smart sensor framework for SHM	x	n/a		Good	Good		High	x	x			Service oriented	Middleware focused on SHM
TinySOA	n/a	n/a		n/a	Good		High	x			x	Service oriented	Semantic-aware routing
Sensei	n/a	n/a		n/a	Good	x	High	x			x	Service oriented	Semantic modeling
Music	x	x		n/a	Good	x	High				x	Service oriented	Adaptation of component-based architectures
SensorsMW	x	x		Good (configurable)	Good		High	x			x	Service oriented	Supports Service Level Agreements
UbiSOAP		x		Dependant on the communication protocol	Good	x (web services)	High					Service oriented	SOAP over different protocols
KASOM	x	n/a		Good	Good	x (web services)	High	x		x		Service oriented	REST web services, Knowledge management functionality
CHOReOS	x	x (dependant on the underlying protocol)		n/a	Good	x	High					Service oriented	It can work over other middleware with different programming models
Servilla	x	n/a	x	Good	Good	x	High	x	x			Service oriented	Focused on heterogeneous WSNs

(*continued on next page*)

Table 8 (*continued*)

Middleware	QoS (Reliability)	QoS (Fault tolerance)	QoS (Real-time)	QoS (Energy efficiency)	QoS (Scalability)	Heterogeneity	Abstraction level	Used in WSN	Used in SHM	Security	Support IoT	Programming model	Features
MidSHM	x	x	x	Good	Good	x	High	x	x			Service oriented	SOAP web services implementation. There is no implementation yet
Maté		n/a		Good	Good	x (TinyOS)	High	x				Based on virtual machines	Bytecode interpreter over TinyOS
MagnetOS	n/a			Good	Good	x	High	x				Based on virtual machines	A distributed OS that abstracts the WSN as a unified Java virtual machine
SwissQM				Good	Good	x	High	x				Based on virtual machines	Bytecode instruction set that is independent of sensor platforms
Smart messages	x	x (dependant on the underlying protocol)	x (Basic)		Good	x	High			x		Agent-based Virtual Machine	Smart messages migrates among nodes relaying on VM
ActorNet	x			Good	Good	x	High	x				Agent-based	Provides services for virtual memory, context switching and multitasking
Agilla	x		x	Good	Good	x (TinyOS)	High	x		x		Agent-based	It combines mobile agents with coordination model
Mobile Agent Computing Paradigm for flexible SHM	x	x	n/a	Bad	Good		High	x	x	n/a		Agent-based	Mobile-C allows agents to move with less bandwidth
TC-Mote/TC-WSANs			-/x	x	Good	x (TinyOS)	High	x				Tuple spaces	Shared channel tuple spaces for communication and synchronization
Cluster-based data aggregation architecture	x	x		Good	Good	x (Java)	High	x	x			Database oriented	Clusters are built depending on node's energy and link quality
GSN		x	x		Good	x	High	x	x	x	x	Database oriented	GSN applications are independent from underlying hardware
SINA		n/a	x	Good	Good		High	x				Database oriented	Provides automatic cluster organization service
COUGAR		n/a		Good	Good	n/a	High	x				Database oriented	Two types of data, sensors and events
MiLAN	x (configurable by user)	x	x	Good	Good	n/a	High	x		n/a	partially	Application oriented	Application and network are unified in a single middleware system
TinyCubus	x (configurable by user)	n/a	x	Not good	Bad	x (TinyOS)	High	x				Application oriented	Adaptative cross-layer framework
MidFusion	x	x	x	x	Good	n/a	High	x				Application oriented	It is applied to applications that need to make a fusion of information

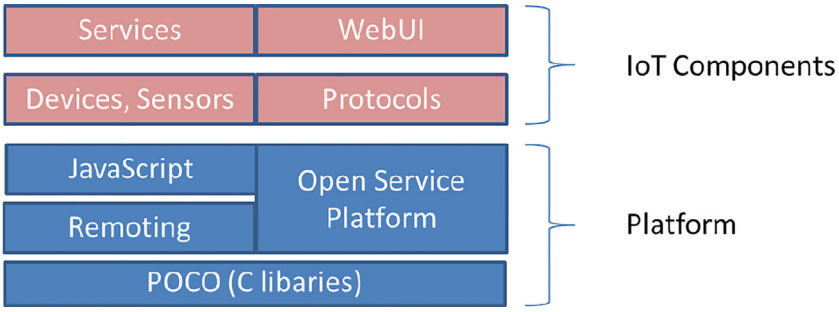

Fig. 8. Macchina.io diagram.

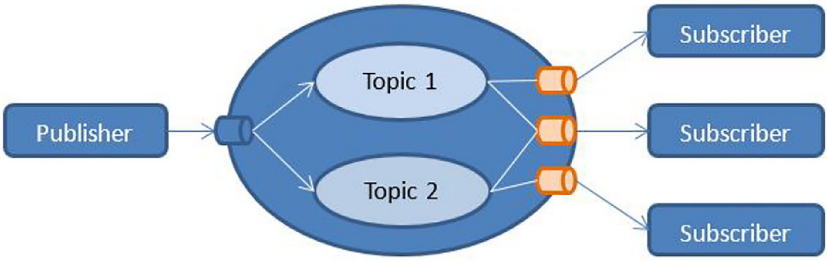

Fig. 9. Publish/subscribe schema.

5.2. Academic middleware

In this section, each of the academic middleware categories for WSNs and the most representative middleware are described. Additionally, emphasis is placed on those middleware that provide QoS features that may be useful for SHM.

5.2.1. Event-based

In event-based middleware, components, applications, and other participants interact through events. Each event has a type, as well as a set of parameters whose values define the state of the event producer. Events travel from producers to subscribers or consumers. A subtype of this type of middleware is Message-oriented middleware (MOM). In this model the communication is based on the messages that contain more information than in the case of the events. Typically, event-based middleware uses a publish / subscribe communication schema. In this communication model a set of subscribers and publishers exist, as shown in Fig. 9. In this way the publishers publish a series of topics that are nothing more than channels that contain certain types of information. These topics are accessible to subscribers who can subscribe to the ones they want. Once the publisher generates a message in a topic automatically, and asynchronously, the message will be sent to each subscriber of that topic. This type of communications design provides QoS requirements such as reliability, availability, real-time performance, scalability and security.

There are several event-based middleware and more particularly publish / subscribe. Hermes[65], Runes[66] or Green[67]. However, they do not consider middleware-level non-functional properties and interoperability between WSNs and access networks. TinyDDS [68] is a middleware that allows interoperability between a sensor network and the access networks. It provides a programming language and protocol interoperability based on the Data Distribution Service (DDS) standard. The TinyDDS framework allows WSN applications to have control over non-functional application and middleware requirements. The results of current simulations and tests indicate that TinyDDS is lightweight and takes up little memory space.

In our opinion event-based middleware is appropriate in those systems where mobility and communications failures are common, as this type of middleware manages it well. This type of middleware is designed to better meet the QoS aspects such as reliability, availability, real-time scalability. Furthermore, TinyDDS provides protocol interoperability, which make it a good candidate for SHM applications. The drawbacks of this middleware are that this type of middleware is not autonomous, and it does not support semantic subscription.

5.2.2. Service oriented

In this case the software or the applications are developed in the form of services that are available to be used (Fig. 10). This type of middleware is based on SOA (Service Oriented Architecture) and is typically used by companies. The main characteristics of this type of technology are under coupling, reusability, composition and service discovery. However, devices with very limited resources can turn discovery services and service composition into a challenge to be overcome.

We now present those which we consider to be the most representative service-oriented middlewares. SensorsMW [69] is an adaptive and flexible service-oriented middleware for QoS configuration and WSN management. A WSN is abstracted as a collection of services in a way that provides complete integration with enterprise information systems. This allows an easy and efficient WSN configuration for the collection of information using Web Services. WSN resources are managed to meet certain QoS requirements, defined in the services. Additionally, an abstract way of accessing resources is offered so that high-level applications can reconfigure and maintain the network throughout the application lifecycle. This middleware does not provide resource discovery. UbiSOAP [70] provides complete integration of the network with Web Services. The architectural resources layer has the necessary functions, including a unified abstraction for simple services (sensors, actuators, processors or software components) to help integrate applications and services with resources. A service support component facilitates the discovery and dynamic composition of resources (eg services). Dynamic composition and instantiation of new services are facilitated by semantic models and descriptions of sensors, actuators and processing elements. The resource layer also contains privacy and security features. Its multi-radio network layer allows the use of heterogeneous networks thanks to a network-independent addressing scheme. This layer also offers functionalities for QoS (energy consumption and availability). CHOREeOS [71] looks for features such as scalability, interoperability, mobility and

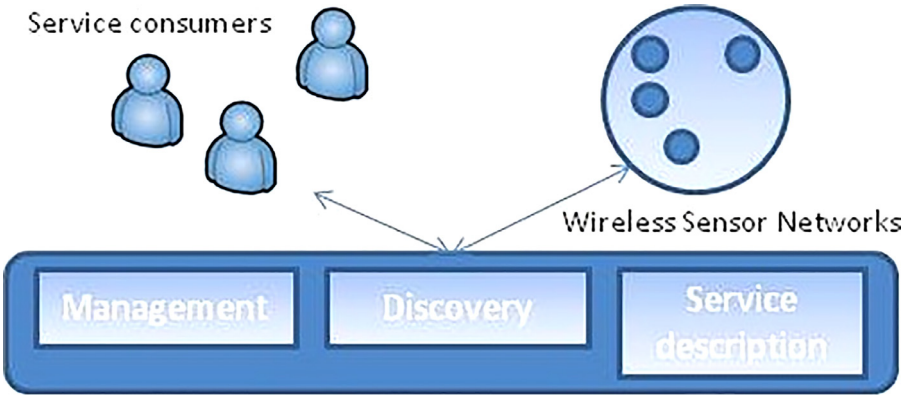

Fig. 10. Service-based middleware schema.

adaptability, is oriented to the IoT and is composed of four components: 1) Executable Services Composition (XSC) to coordinate the composition of services; 2) extensible access to services (XSA); 3) Extensible Services Discovery (XSD) to manage protocols and processes for service discovery, and 4) cloud and mesh middleware to manage computing resources and direct the development of choreography. MidSHM [10] is a service-oriented middleware, based on SOAP, which pursues eight basic requirements in SHM applications which are: resource optimization, dynamic network topology, in-network processing, QoS, heterogeneity, fault tolerance, real-word awareness and runtime reconfiguration. In order to accomplish them this middleware has three layers: Network, Task management and Application management. This looks like a promising middleware for SHM applications, however to the best of our knowledge it is still a work in progress. On the other hand, SOAP web services implementation is based on standards like WSDL (Web Services Description Language), which are not appropriate for resource-constrained devices.

Apart from those we have just described, there is a large set of other service-oriented middlewares such as Servilla [72], TinySOA [73], SENSEI [74], MUSIC [75] and KASOM [76].

For all these solutions, the definition of services can be extracted for components of the middleware, especially in environments where it is necessary to discover who will provide what services, and which can be composed among them to give rise to other services. However in this case, aspects of QoS are not a priority. In [77] a flexible smart sensor framework for SHM is presented. The middleware runs over the Imote2 sensor platform and uses the open source libraries provided by the Illinois Structural Health Monitoring Project [57] (ISHMP) to provide a middleware for WSNs with functionality for SHM such as sleep cycling and threshold triggering. The middleware presented has been validated in a real deployment monitoring the structural health of a cable-stayed bridge in the Jindo islands (South Korea). The ISHMP offers a set of services that can be used to implement SHM algorithms for modal analysis and damage detection and services.

In general, service-oriented middlewares can be successfully applied as presented in [10,77]. However, we consider that WSDL might be adapted to resource-constrained devices. By doing so, this kind of middleware can provide highly configurable QoS which is a very important requirement for SHM.

5.2.3. Based on virtual machines

Virtual machine-based middleware (Fig. 11) provides a secure execution environment for user applications, thanks to infrastructure virtualization. In this way, applications are divided into separate modules that are distributed over the network. Each node in the network uses a virtual machine (VM). These types of solutions provide a high level of abstraction, self-management and adaptability. This type of middleware can be further divided into two types: a) at the application level (VMs

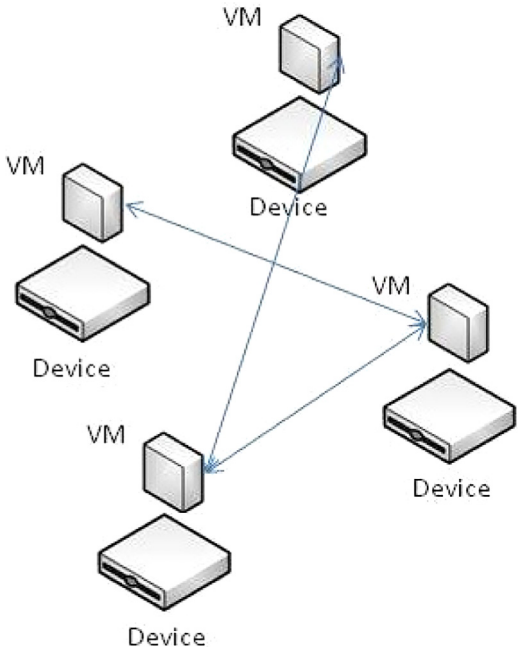

Fig. 11. VM-based middleware schema.

are located between the OS and the applications) and b) at the system level (they replace the entire OS).

Some of the most relevant implementations are Maté [78], MagnetOS [79] or SwissQM [80]. However, this type of middleware is not suitable for applications based on events that require non-blocking operations. On the other hand, performance such as reliability (particularly on constrained resources) is not one of the strong points. We therefore consider this kind of middleware to be unsuitable for SHM.

5.2.4. Agent-based

In agent-based middleware (Fig. 12), the applications are divided into modular programs that facilitate the injection and distribution through the network thanks to the mobile agents. As agents migrate from one node to another, they maintain their execution state. This facilitates the design of decentralized systems useful for fault tolerance of the network.

Smart Messages [81] proposes an autonomous network architecture for large-scale systems of embedded nodes, which are limited in resources, heterogeneous and volatile. Smart Messages overcomes these node constraints by migrating agents to nodes of interest, using application-driven routing instead of point-to-point communication between nodes. The main contribution of this middleware is the high flex-

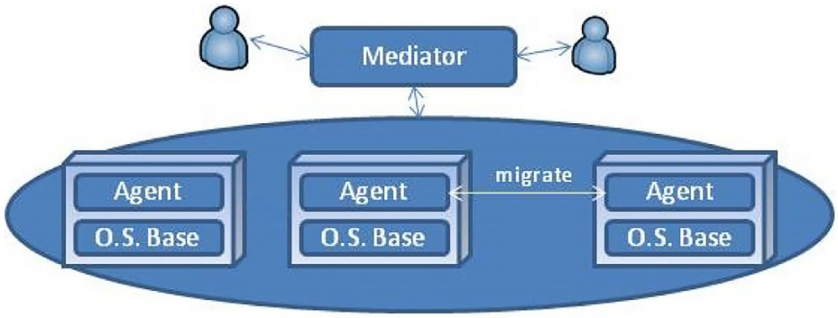

Fig. 12. Agent-based middleware schema.

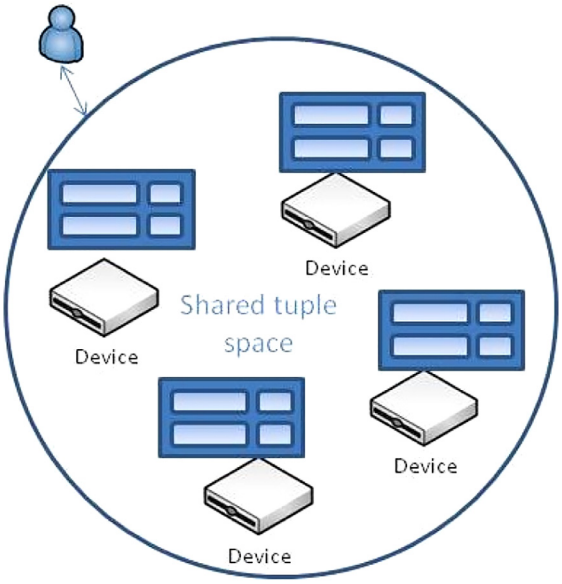

Fig. 13. Tuple space-based middleware schema.

ibility it provides in the case of dynamic network configurations. However, Smart Messages does not support multiple applications. Moreover, it only considers nodes with limited resources. Mobile Agent Computing Paradigm for Building a Flexible SHM Sensor Network [82] objective is to reduce data transmission and improve flexibility of SHM systems. The Mobile-C language is used by the authors to achieve mobile agent generation, migration, execution and management with less bandwidth. They define two types of agents, stationary (staying in the sensor nodes where they were created) and mobile (created during systems's operation and are able to move to different sensor nodes). This middleware assumes that each sensor node is powerful and also there is no concern about how each sensor node might be powered.

ActorNet [83] and Agilla [84] are also relevant agent-based middlewares.

These solutions do not guarantee the real-time feature, at least in the harder case, although SHM applications do not need hard real-time. The autonomous characteristic of the agents can lead the execution of the system to an unpredictable point. This kind of behavior is not admissible in critical systems like SHM. Moreover, mobile agents are susceptible to message loss, especially in environments where nodes have limited resources. Therefore, we do not consider agent-based middleware to be valid middleware for SHM.

5.2.5. Tuple spaces

Tuple space middleware (Fig. 13) is middleware based on coordination languages, and in particular, those based on Linda. They are characterized by the fact that each node maintains a structure of local tuples. After all, a tuple space is a data repository that can be accessed concurrently. In this way nodes communicate by writing tuples in the shared space.

TC-Mote [85], TC-WSANs [86] are tuple-space-based middleware, where each is applied to a specific environment. TC-Mote is a middleware based on a coordination language, oriented to WSNs, taking Linda as a reference, although in this case it is based on a channel space of tuples. A tuple channel is a FIFO structure that allows for a variety of one-to-many and many-to-one modes of communication, through which tuples travel. Additionally, a series of regions are defined in which there are a number of nodes that share the channel space of tuples. Within each region there is a leader. TC-WSANs is an evolution of TC-Mote oriented to WSAN, and in which real-time characteristics are introduced, being able to assign priorities to two levels, in the tuples's channels as well as in the tuples, so that the tuples of higher priority have preference.

This type of middleware copes well with the problem of protocol interoperability and also with frequent disconnection of nodes and improves asynchronous communication. However, the need to maintain a shared copy of the tuple space on each node is a concern in resource-constrained devices. Overall, its scope is considered suitable for SHM although it should be adapted in order to fit into the SHM requirements.

5.2.6. Database oriented

In database-oriented middleware (Fig. 14), a WSN abstracts itself as if it were a virtual, relational database, so that an application can perform queries using a syntax in SQL language, allowing complex queries to be performed.

The most popular are GSN [87], SINA [88] and COUGAR [89], although there are other database-oriented middleware. All of them, as mentioned, offer an SQL syntax to express queries for the network.

However, this type of middleware has a few drawbacks which makes it unsuitable for SHM applications as this type of middleware does not support real-time requirements well, nor does it meet energy-aware requirements. In general, this kind of middleware is not often used in critical system applications. Moreover, these types of middleware use a centralized communication model in a way that complicates the management of networks with large numbers of nodes, or dynamic networks, with nodes constantly appearing.

5.2.7. Application specifics

Middleware of this type relies on supporting resource management (QoS) for a specific application by implementing an architecture that fits well with the domain requirements of the application in question. MiLAN [90] is an application-specific middleware. The applications that are deployed on it specify their QoS requirements and the middleware automatically adapts the network configuration to best match the requirements of the application while minimizing power consumption. The application specifies its needs through state-based requirement variables and sensor QoS graphs. These two mechanisms allow the application to define its interest variables and QoS requirements for each.

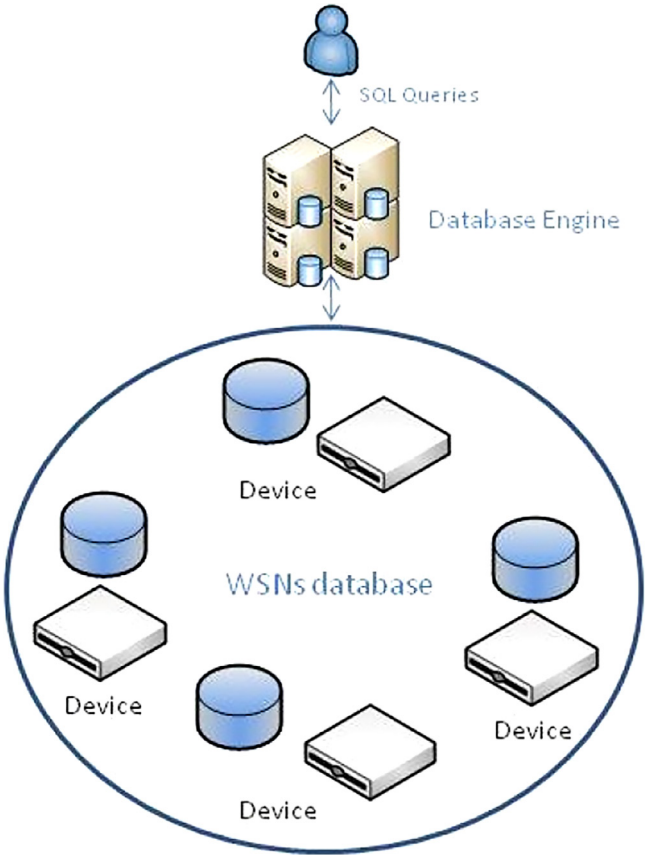

Fig. 14. Database-based middleware schema.

To improve its performance, the architecture of Milan extends into the layer of network protocol. TinyCubus [91] is a cross-layer middleware that also runs on TinyOS. It proposes a generic, extensible and flexible framework that is able to adapt to new application requirements. Specific application requirements are met by customizing the generic components. However, the fact that the design is "cross-layer" causes an overload to be introduced that ultimately effects the power management of the devices. Additionally, this solution is not scalable, as initially, TinyCubus was designed to monitor the structural health of bridges and for driving support systems. MidFusion [92] is constructed based on the MiLAN concepts. The aim of this middleware is to avoid having to know the list of available sensors thanks to the use of a Bayesian network and decision theory to provide a portable abstraction of the application infrastructure. MidFusion was designed with the objective of applying it to applications that need to fuse information (e.g., intrusion detection system).

Due to particular features and requirements of SHM middleware, where QoS parameters highly important, application specific middleware could be a promising option. However, we consider other options like publish / subscribe or service-oriented middleware to be more suitable for SHM applications.

6. Conclusions

CI monitoring can have unique characteristics, depending on the type of monitoring to be performed. Based on the type of monitoring and the location of sensors (easy access/remote), a given type of sensor and communication technology will be deployed, taking data with a certain periodicity (eventual / continuous). From the point of view of the communications, a wireless communication protocol will be used, depending on several aspects, such as the energy consumed, the communication distance, the sending rate and environment. Current communication protocols follow two differentiated approaches. The first uses gateway or sink nodes to collect sensor data about a group of sensor nodes and the second relies on independent nodes, each of them sending information directly to the cloud. We have identified novel technologies such as LTE-M, Bluetooth LE, LoRaWAN and SIGFOX as promising in this field. It is worth noting that there is increasing interest in technologies that can deliver data directly to the cloud such as SIGFOX or LoRaWAN.

There are presently a large number of companies that supply sensor devices for the SHM of CIs and although most deployments are carried out in the context of research projects the number of deployments using this technology is increasing at a steady pace.

Dealing with and programming WSNs can be complex task for programmers and also for maintenance staff. For this reason, the use of middleware that abstracts WSN complexity at both the programming and maintenance stages is very important. Therefore, in this paper we have presented the main challenges of WSN-based solutions for SHM and we have also defined the requirements that WSN middleware for SHM applications should pursue. With these requirements in mind, the most representative middleware have been surveyed and their suitability for specified requirements has been studied, considering their main benefits and drawbacks. In conclusion, the middleware we consider the most promising for SHM applications are the following: AllJoyn as it is open-source, designed to be used on any kind of device and it considers interoperability and QoS, however the middleware should be customized for SHM. Event-based middleware, in particular publish / subscribe, are very useful for SHM middleware as they provide protocol and data interoperability, they also manage mobility and communications failure well, making the WSN robust and reliable, however the QoS configuration layer needs to be improved. Another interesting category is service-oriented middleware, as they can provide configurable QoS layer and interoperability, however, its WSDL should be redefined and simplified so resource-constrained devices can manage service contracts easily. Tuple spaces middleware provide protocol abstraction enabling interoperability, and the fact that they provide fault-tolerance of nodes is a plus. However, on the downside, maintaining local copies of tuple channels on nodes might not be suitable for resource-constrained devices.

Contrarily, we consider database, agent and virtual machines oriented unsuitable for SHM applications due to their lack on interoperability, reliability, fault-tolerance, and because battery life of constrained resources might be shortened due to the overhead introduced by these middleware.

Acknowledgments

This work has been funded by the Spanish projects TIC-1572 ("MIsTIca: Critical Infrastructures Monitoring based on Wireless Technologies") and RTC-2016-5164-8 ("KAMIC Development of a Self-Installable Kit for Structural Monitoring of Critical Infrastructures").

References

[1] C.R. Farrar, K. Worden, An introduction to structural health monitoring, Philos. Trans. R. Soc. London A 365 (1851) (2007) 303–315.
[2] I. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, Wireless sensor networks: a survey, Comput. Netw. 38 (4) (2002) 393–422.
[3] J. Chen, M. Díaz, L. Llopis, B. Rubio, J.M. Troya, A survey on quality of service support in wireless sensor and actor networks: requirements and challenges in the context of critical infrastructure protection, J. Netw. Comput. Appl. 34 (4) (2011) 1225–1239. Advanced Topics in Cloud Computing.
[4] Wsan4cip. wireless sensor and actuator networks for the protection of critical infrastructures. eu fp7, (http://www.wsan4cip.eu).
[5] Fastrack project. eco-friendly and sustainable slab track for high-speed lines. federinnterconecta 2013, (http://www.fastrack.es).
[6] Ps-quasar. a dependable middleware for the development of applications for wsans, (http://www.lcc.uma.es/˜tolo/psquasar/index.html).
[7] S. Hadim, N. Mohamed, Middleware: middleware challenges and approaches for wireless sensor networks, IEEE Distrib. Syst. Online 7 (3) (2006). 1–1.
[8] L. Mottola, G.P. Picco, Programming wireless sensor networks: fundamental concepts and state of the art, ACM Comput. Surv. 43 (3) (2011) 19:1–19:51.

[9] B. Bhuyan, H.K.D. Sharma, N. Sharma, A survey on middleware for wireless sensor networks, J. Wireless Netw. Commun. 4 (1) (2014) 7–17.
[10] Y. Sahni, J. Cao, X. Liu, MidSHM: a flexible middleware for wsn-based SHM application using service-oriented architecture, in: IEEE Symposium on Service-Oriented System Engineering (SOSE), 2016, Oxford, 2016, pp. 126–135.
[11] T.C. Arcadius, B. Gao, G. Tian, Y. Yan, Structural health monitoring framework based on internet of things: a survey, IEEE Internet Things J. PP (99) (2017). 1–1.
[12] E. Cañete, J. Chen, M. Díaz, L. Llopis, B. Rubio, Sensor4pri: a sensor platform for the protection of railway infrastructures, Sensors 15 (3) (2015) 4996–5019.
[13] E. Cañete, J. Chen, M. Díaz, L. Llopis, A. Reyna, B. Rubio, Using wireless sensor networks and trains as data mules to monitor slab track infrastructures, Sensors 15 (7) (2015) 15101–15126.
[14] KAMIC, Development of a self-installable kit for structural monitoring of critical infrastructures, spanish project. reference number rtc-2016-5164-8, (http://www.cemosa.es/proyecto-idi/kamic-development-of-self-installable-kit-for-structural-monitoring-of-critical-infrastructures).
[15] S. Wijetunge, U. Gunawardana, R. Liyanapathirana, Wireless sensor networks for structural health monitoring: considerations for communication protocol design, in: 2010 17th International Conference on Telecommunications, 2010, pp. 694–699.
[16] P. Wang, Y. Yan, G.Y. Tian, O. Bouzid, Z. Ding, Investigation of wireless sensor networks for structural health monitoring, J. Sensors 2012 (2012) (2012) 1–7.
[17] Y. Lim, G. Ferrari, H. Takahashi, M. Montn, Device-to-device communications in wireless sensor networks, Int. J. Distrib. Sensor Netw. 12 (8) (2016). 1550147716664233.
[18] IEEE, 802.15.4 specification, (http://standards.ieee.org/about/get/802/802.15.html).
[19] WirelessHART, HART Communication Foundation, http://www.hartcomm.org.
[20] ZigBee, ZigBee Alliance, http://www.zigbee.org.
[21] DASH7 Alliance, Wireless sensor and actuator protocol. dash7 alliance std. rev. v1.0., (http://www.dash7-alliance.org/product/dash7-alliance-protocol-specification-v1-0/).
[22] G. Ergeerts, M. Nikodem, D. Subotic, T. Surmacz, B. Wojciechowski, P.D. Meulenaere, M. Weyn, Dash7 alliance protocol in monitoring applications, in: 2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2015, pp. 623–628.
[23] S. Barrachina, B. Bellalta, T. Adame, A. Bel, Multi-hop communication in the uplink for lpwans, CoRR abs/1611.08703(2016).
[24] 6LoWPAN, IETF, http://www.ietf.org/dyn/wg/charter/6lowpan-charter.html.
[25] R. Sanchez-Iborra, M.-D. Cano, State of the art in lp-wan solutions for industrial iot services, Sensors 16 (5) (2016).
[26] C. Alcaraz, S. Zeadally, Critical infrastructure protection: requirements and challenges for the 21st century, Int. J. Crit. Infrastruct. Prot. 8 (2015) 53–66.
[27] ISA, International Society of Automation, ISA100.11.a specification: wireless systems for industrial automation: process control and related applications, www.isa.org/isa100.
[28] ISA100, Architecture for industrial internet of things (brochure), (http://isa100wci.org/en-us/documents/pdf/3405-isa100-wirelesssystems-future-broch-web-etsi).
[29] J.-H. Lee, J.-E. Jung, N.-G. Kim, B.-H. Song, Industrial pipe-rack health monitoring system based on reliable-secure wireless sensor network, Int. J. Distrib. Sensor Networks 8 (10) (2012) 641391.
[30] SIGFOX, https://www.sigfox.com/.
[31] A. Saifullah, M. Rahman, D. Ismail, C. Lu, R. Chandra, J. Liu, Snow: sensor network over white spaces, in: Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM, in: SenSys '16, ACM, New York, NY, USA, 2016, pp. 272–285.
[32] LTE-M, 3rd generation partnership project (3gpp) lte-m communication protocol, (http://www.3gpp.org/technologies/keywords-acronyms/98-lte).
[33] A. Saifullah, M. Rahman, D. Ismail, C. Lu, R. Chandra, J. Liu, Snow: sensor network over white spaces, in: Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM, in: SenSys '16, ACM, New York, NY, USA, 2016, pp. 272–285.
[34] Weightless, http://www.weightless.org/.
[35] S. Lawson, Narrowband iot standard for machines moves forward, (http://www.computerworld.com/article/2984928/mobile-wireless/narrowband-iot-standard-for-machines-moves-forward.html).
[36] D. Jones, Ericsson, intel, nokia back new narrowband lte iot spec, (http://www.lightreading.com/mobile/4g-lte/ericsson-intel-nokia-back-new-narrowband-lte-iot-spec-/d/d-id/718162).
[37] I. Scales, 3gpp agrees harmonized proposal for narrowband iot radio technology, (http://www.telecomtv.com/articles/iot/3gpp-agrees-harmonized-proposal-for-narrowband-iot-radio-technology-12853/).
[38] nwave.io, nwave communication protocol, (http://www.nwave.io).
[39] L. Alliance, Lorawan communication protocol, (https://www.lora-alliance.org/).
[40] A. Augustin, J. Yi, T. Clausen, W.M. Townsley, A study of lora: long range & low power networks for the internet of things, Sensors 16 (9) (2016).
[41] I. Bluetooth SIG, Bluetooth communication protocol, (https://www.bluetooth.com/).
[42] theinternetofthings.report, Rpma technology for the internet of things, ingenu tech. 2016.
[43] Plum, Future use of licence exempt radio spectrum, 2015, (http://www.plumconsulting.co.uk/pdfs/Plum_July_2015_Future_use_of_Licence_Exempt_Radio_Spectrum.pdf).
[44] T. Myers, D. Werner, K. Sinsuan, J. Wilson, S. Reuland, P. Singler, M. Huovila, Light monitoring system using a random phase multiple access system, 2013.
[45] RESENSYS, (http://www.resensys.com).
[46] LORD, Microstrain, (http://www.microstrain.com).
[47] SensorCloud, (http://sensorcloud.com).
[48] MONNIT, (http://www.monnit.com).
[49] Beanair, (http://www.beanair.com).
[50] SENSEOR, (http://www.senseor.com).
[51] Libelium, (http://www.libelium.com).
[52] X.d. Jiang, Y.l. Tang, Y. Lei, Wireless sensor networks in structural health monitoring based on zigbee technology, in: 2009 3rd International Conference on Anti-counterfeiting, Security, and Identification in Communication, 2009, pp. 449–452.
[53] H. Khaleel, F. Penna, C. Pastrone, R. Tomasi, M.A. Spirito, Frequency agile wireless sensor networks: design and implementation, IEEE Sensors J. 12 (5) (2012) 1599–1608.
[54] P.J. Bennett, K. Soga, I. Wassell, P. Fidler, K. Abe, Y. Kobayashi, M. Vanicek, Wireless sensor networks for underground railway applications: case studies in prague and london, Smart Struct. Syst. 6 (5–6) (2010) 619–639.
[55] S. Jang, H. Jo, S. Cho, K. Mechitov, J. Rice, S. Sim, H. Jung, C. Yun, B. Spencer, G. Agha, Structural health monitoring of a cable-stayed bridge using smart sensor technology: deployment and evaluation, Smart Struct. Syst. 6 (5–6) (2010) 439–459.
[56] V. Gattulli, F. Graziosi, F. Federici, F. Potenza, A. Colarieti, M. Lepidi, Structural health monitoring of the basilica s. maria di collemaggio, 2013.
[57] Illinois SHM, Full-scale applications of wsn, (http://shm.cs.uiuc.edu/Full%20scale%20applications.html).
[58] M.A. Razzaque, M. Milojevic-Jevric, A. Palade, S. Clarke, Middleware for internet of things: a survey, IEEE Internet Things J. 3 (1) (2016) 70–95.
[59] B. Rubio, M. Diaz, J.M. Troya, Programming approaches and challenges for wireless sensor networks, in: 2007 Second International Conference on Systems and Networks Communications (ICSNC 2007), 2007. 36–36
[60] E. Kura, (http://www.eclipse.org/kura).
[61] MQTT, (http://www.mqtt.org).
[62] U. Hunkeler, H.L. Truong, A. Stanford-Clark, Mqtt-s - a publish/subscribe protocol for wireless sensor networks, in: Communication Systems Software and Middleware and Workshops, 2008. COMSWARE 2008. 3rd International Conference on, 2008, pp. 791–798.
[63] AllJoyn Framework, (https://allseenalliance.org/framework).
[64] macchina.io, (https://macchina.io).
[65] P.R. Pietzuch, J.M. Bacon, Hermes: a distributed event-based middleware architecture, in: Proceedings 22nd International Conference on Distributed Computing Systems Workshops, 2002, pp. 611–618.
[66] P. Costa, G. Coulson, R. Gold, M. Lad, C. Mascolo, L. Mottola, G.P. Picco, T. Sivaharan, N. Weerasinghe, S. Zachariadis, The runes middleware for networked embedded systems and its application in a disaster management scenario, in: Fifth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom'07), 2007, pp. 69–78.
[67] T. Sivaharan, G. Blair, G. Coulson, GREEN: A Configurable and Re-configurable Publish-Subscribe Middleware for Pervasive Computing, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 732–749.
[68] P. Boonma, J. Suzuki, Tinydds: an interoperable and configurable publish/subscribe middleware for wireless sensor networks, in: Principles and Applications of Distributed Event-Based Systems, IGI Global, 2010, pp. 206–231.
[69] G.F. Anastasi, E. Bini, A. Romano, G. Lipari, A service-oriented architecture for qos configuration and management of wireless sensor networks, in: 2010 IEEE 15th Conference on Emerging Technologies Factory Automation (ETFA 2010), 2010, pp. 1–8.
[70] M. Caporuscio, P.G. Raverdy, V. Issarny, ubisoap: a service-oriented middleware for ubiquitous networking, IEEE Trans. Serv. Comput. 5 (1) (2012) 86–98.
[71] A. Ben Hamida, F. Kon, N. Lago, A. Zarras, D. Athanasopoulos, D. Pilios, P. Vassiliadis, N. Georgantas, V. Issarny, G. Mathioudakis, G. Bouloukakis, Y. Jarma, S. Hachem, A. Pathak, Integrated CHOReOS Middleware - Enabling Large-scale, QoS-Aware Adaptive Choreographies, 2013. Working paper or preprint
[72] C.-L. Fok, G.-C. Roman, C. Lu, Servilla: a flexible service provisioning middleware for heterogeneous sensor networks, Sci. Comput. Programming 77 (6) (2012) 663–684. (1) Coordination 2009 (2) WCRE 2009
[73] E. Avilés-López, J.A. García-Macías, Tinysoa: a service-oriented architecture for wireless sensor networks, Serv. Oriented Comput. Appl. 3 (2) (2009) 99–108.
[74] M. Presser, P.M. Barnaghi, M. Eurich, C. Villalonga, The sensei project: integrating the physical world with the digital world of the network of the future, IEEE Commun. Mag. 47 (4) (2009) 1–4.
[75] R. Rouvoy, P. Barone, Y. Ding, F. Eliassen, S. Hallsteinsen, J. Lorenzo, A. Mamelli, U. Scholz, MUSIC: Middleware Support for Self-Adaptation in Ubiquitous and Service-Oriented Environments, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 164–182.
[76] I. Corredor, J.F. Martínez, M.S. Familiar, L. López, Knowledge-aware and service-oriented middleware for deploying pervasive services, J. Netw. Comput. Appl. 35 (2) (2012) 562–576.
[77] J. Rice, K. Mechitov, S. Sim, T. Nagayama, S. Jang, R. Kim, B. Spencer, G. Agha, Y. Fujino, Flexible smart sensor framework for autonomous structural health monitoring, Smart Struct. Syst. 6 (5–6) (2010) 423–438.
[78] P. Levis, D. Culler, MatÉ: a tiny virtual machine for sensor networks, SIGARCH Comput. Archit. News 30 (5) (2002) 85–95.
[79] R. Barr, J.C. Bicket, D.S. Dantas, B. Du, T.D. Kim, B. Zhou, E.G. Sirer, K. Bing, Z. Emin, G. Sirer, On the need for system-level support for ad hoc and sensor networks, Oper. Syst. Rev. 36 (2002) 1–5.
[80] R. Mueller, G. Alonso, D. Kossmann, Swissqm: next generation data processing in sensor networks, CIDR07, 2007.
[81] P. Kang, C. Borcea, G. Xu, A. Saxena, U. Kremer, L. Iftode, Smart messages: a distributed computing platform for networks of embedded systems, Comput. J. 47 (4) (2004) 475.

[82] B. Chen, W. Liu, Mobile agent computing paradigm for building a flexible structural health monitoring sensor network, Comput.-Aided Civil Infrastruct. Eng. 25 (7) (2010) 504–516.
[83] Y. Kwon, S. Sundresh, K. Mechitov, G. Agha, Actornet: an actor platform for wireless sensor networks, in: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems, in: AAMAS '06, ACM, New York, NY, USA, 2006, pp. 1297–1300.
[84] C.-L. Fok, G.C. Roman, C. Lu, Rapid development and flexible deployment of adaptive wireless sensor network applications, in: 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05), 2005, pp. 653–662.
[85] M. Diaz, B. Rubio, J.M. Troya, Tcmote: a tuple channel coordination model for wireless sensor networks, in: ICPS '05. Proceedings. International Conference on Pervasive Services, 2005., 2005, pp. 437–440.
[86] J. Barbaran, M. Diaz, I. Esteve, D. Garrido, L. Llopis, B. Rubio, J.M. Troya, Tc-wsans: a tuple channel based coordination model for wireless sensor and actor networks, in: 2007 12th IEEE Symposium on Computers and Communications, 2007, pp. 173–178.
[87] K. Aberer, M. Hauswirth, A. Salehi, A middleware for fast and flexible sensor network deployment, in: Proceedings of the 32Nd International Conference on Very Large Data Bases, in: VLDB '06, VLDB Endowment, 2006, pp. 1199–1202.
[88] C.-C. Shen, C. Srisathapornphat, C. Jaikaeo, Sensor information networking architecture and applications, IEEE Pers. Commun. 8 (4) (2001) 52–59.
[89] P. Bonnet, J. Gehrke, P. Seshadri, Towards sensor database systems, in: Proceedings of the Second International Conference on Mobile Data Management, in: MDM '01, Springer-Verlag, London, UK, UK, 2001, pp. 3–14.
[90] W.B. Heinzelman, A.L. Murphy, H.S. Carvalho, M.A. Perillo, Middleware to support sensor network applications, IEEE Netw. 18 (1) (2004) 6–14.
[91] P.J. Marron, A. Lachenmann, D. Minder, J. Hahner, R. Sauter, K. Rothermel, Tinycubus: a flexible and adaptive framework sensor networks, in: Proceeedings of the Second European Workshop on Wireless Sensor Networks, 2005., 2005, pp. 278–289.
[92] H. Alex, M. Kumar, B. Shirazi, Midfusion: an adaptive middleware for information fusion in sensor network applications, Inf. Fusion 9 (3) (2008) 332–343.