Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 212876, 18 pages
doi:10.1155/2012/212876

*Research Article*

# Wireless Communication Networks for Gas Turbine Engine Testing

**Xuewu Dai,[1] Konstantinos Sasloglou,[2] Robert Atkinson,[2] John Strong,[3] Isabella Panella,[3] Lim Yun Cai,[4] Han Mingding,[4] Ang Chee Wei,[4] Ian Glover,[2] John E. Mitchell,[1] Werner Schiffers,[5] and Partha S. Dutta[6]**

[1] *Department of Electronic and Electrical Engineering, University College London, London WC1E 7JE, UK*
[2] *Department of Electronic and Electrical Engineering, University of Strathclyde, Glasgow G1 1XW, UK*
[3] *SELEX Galileo, Basildon SS14 3EL, UK*
[4] *Institute for Infocomm Research, A*STAR, Singapore*
[5] *Strategic Research Centre, Rolls-Royce plc, P.O. Box 31, Derby DE24 8BJ, UK*
[6] *Advanced Technology Centre, Rolls-Royce Singapore Pte Ltd., 16 International Business Park, No. 03-01 M+W Zander, Singapore 609929*

Correspondence should be addressed to John E. Mitchell, jmitchel@ee.ucl.ac.uk

Received 1 June 2011; Revised 28 September 2011; Accepted 20 October 2011

Academic Editor: Frank Ehlers



A new trend in the field of Aeronautical Engine Health Monitoring is the implementation of wireless sensor networks (WSNs) for data acquisition and condition monitoring to partially replace heavy and complex wiring harnesses, which limit the versatility of the monitoring process as well as creating practical deployment issues. Augmenting wired with wireless technologies will fuel opportunities for reduced cabling, faster sensor and network deployment, increased data acquisition flexibility, and reduced cable maintenance costs. However, embedding wireless technology into an aero engine (even in the ground testing application considered here) presents some very significant challenges, for example, a harsh environment with a complex RF transmission channel, high sensor density, and high data rate. In this paper we discuss the results of the Wireless Data Acquisition in Gas Turbine Engine Testing (WIDAGATE) project, which aimed to design and simulate such a network to estimate network performance and derisk the wireless techniques before the deployment.

## 1. Introduction

Wireless sensors are increasingly used for monitoring structures and machinery. A large number of such systems exist already on the market [1]. Most systems comprise a relatively small number of nodes with low date rates; however, there are clear signs that wireless sensor technology is maturing [2]. The work described in this document explores a wireless sensor system for monitoring vital parameters during aero gas turbine engine development tests with a long-term aim to do the same during engine on wing operation. A typical engine test phase requires measurements of up to 3000 parameters from transducers on the engine connected to the data acquisition system through very long cables. These wired data acquisition systems require as much as 12 km of wiring and involve long and expensive setup and instrumentation times which significantly increases time to market.

Despite these limitations, wired instrumentation is a mature and well-understood approach used widely in the aero industry. Replacing it with wireless solutions will require significant changes in not only the technology but also in the associated engineering processes. In the absence of sufficient know-how about the performances of wireless sensors for engine test data acquisition, replacing the existing instrumentation process is fraught with risks. In this context, the WIDAGATE project has developed robust and experimentally validated simulations of WSNs to generate insights into their performance for engine testing applications. This project aims to provide the aero engine testing industry the tools to conduct an effective risk-benefit trade-off analysis and

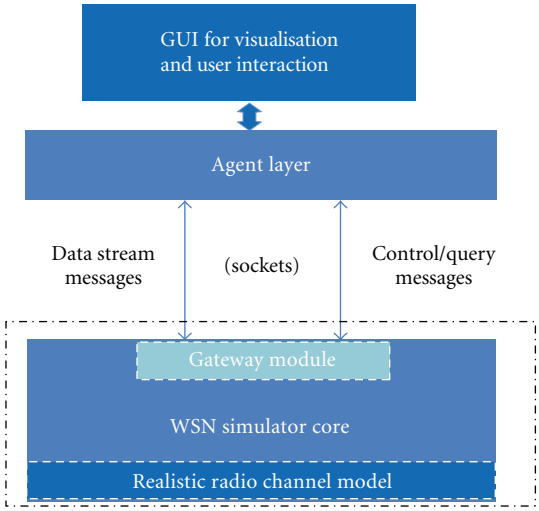



Figure 1: System structure of the software simulator developed.

support intelligent investment choices regarding WSN-based instrumentation.

Specific advantages of using WSN-based instrumentation are in the replacement of part of the wiring infrastructure with wireless communication to offer significant benefits in cost and time, flexibility, interoperability, weight, and improved robustness. To achieve these goals a number of long-term challenges need to be addressed, in particular, the issue of communicating in the harsh and dynamic environment of gas turbines involving high-speed rotations, rapid airflows, high temperatures, and large-amplitude vibrations. In terms of wireless communication, the WSNs in the application of engine testing face four challenges:

(1) severe RF interference;

(2) non-line-of-sight propagation. RF signals are transmitted in an environment that is largely composed of metal and it is highly possible that nodes are not in line-of-sight;

(3) high-density sensors. Engine testing may eventually involve up to a few thousands sensors;

(4) high-data-rate and near real-time transmission requirement with accurate synchronisation.

The WIDAGATE project delivers an application-specific diagnostic tool for network performance, network architecture, and communication protocols analysis in a relatively short time scale (i.e., one run of engine testing), whilst also addressing many generic, long-term WSN research challenges. The main achievement of the WIDGAGTE project is the development of both an accurate and experimentally validated simulation model and a system demonstrator of a wireless sensor network for data gathering and health monitoring during gas turbine engine testing.

As shown in Figure 1, the developed software simulation platform is comprised of three parts: the realistic radio channel model, the wireless network simulator core, and the Agent layer for user interaction and optimisation. The impacts of the complex engine testing environment on wireless communication are modelled by an empirical radio channel model. The radio channel model is integrated into an event-based Wireless Sensor Network (WSN) simulator core. The Agent layer provides the friendly graphical user interface to visualise the network performance, allows users to both access and control the underlying simulator core on-the-fly (e.g., querying sensor nodes for information, change the network architecture by the repositioning of nodes, etc.), and performs multiagent optimisation to improve network performances (such as network throughput).

This paper is organised as follows: Section 2 gives details of the application considered within the WIDAGATE project and details related work. Section 3 gives an overview of the WIDAGATE system design. Section 4 presents details of the modelling of the radio channel in engine testing environment. The Medium Access Control (MAC) protocols and the software simulator of WSN are described in Section 5, followed by Section 6 presenting the design and implementation of a multiagent application layer which allows end-users to interact and control the developed WSN simulator and the optimisation of network performance. Section 7 presents the development of a hardware evaluation test bed and the results of both lab tests and engine tests, with Section 8 presenting the conclusions of the work.

## 2. Application Scenario

Data sampling and transmission of the samples are key issues when developing a new instrumentation system. Generally, there are two kinds of schemes for data sampling and transmission. (1) Offline transmission: the measurement data are sampled and stored at the sensor nodes during the course of engine testing and transmitted to the data logger (referred to as data SINK or data concentrator) at the end of the testing. The advantage of this two-step scheme is the simplicity of the communication system while a disadvantage is the large storage requirement and non-real-time operation. (2) Online transmission: the sensor nodes sample the physical signals and immediately transmit the sampled data through the wireless network. Although the online method can reduce the storage required at each sensor node, it puts a real-time requirement of the communication network, demanding higher throughput and lower latency.

The features of the engine testing and the requirements to the wireless communication system are listed as follows.

*Periodic Traffic Load.* The periodic sensor measurements generate a periodic data flow from the sensors to the collectors. This implies a schedule-based MAC approach, in order to effectively exploit this pattern to maximise performance. Some spatial correlation between the sensor measurements is expected. The MAC protocol must have the ability to manage the local data communication in a manner that enables the available data redundancy to be exploited.

*Near Real-Time Requirement/Latency Requirement.* This project does not attempt to provide real-time operational

Table 1: Environmental conditions during development and production testing.

| | |
|---|---|
| Oil system temperature | 250°C |
| Air temperature (beneath core cover) | 350°C |
| Metal temperature | 1100–1300°C |
| Pressure | 40–42 bar |
| Vibration | 40 g |

data; however, rapid delivery of results and timing accuracy of the data are vital for acceptable operation.

*High Sensor Density.* During engine development testing, there are over three thousands sensors (1000 thermocouples, 1500 pneumatic lines, and 500 accelerometers) required to measure and record the temperature, pressure, and vibration, respectively. Most of the sensors are deployed in the limited space around or within the engine which leads to a high density of sensor points.

*High Data Rate and High Spectral Efficiency MAC Protocol.* The periodically generated sensor data and high density of sensors result in a huge amount of data to be transmitted across the network. Thus, a high data rate is necessary to achieve near real-time and low latency operation. Maximising the system spectral efficiency (throughput per unit bandwidth per unit area) in the multihop sensor network is essential to minimising latency and maximising energy efficiency. Therefore, a high system spectral efficiency MAC protocol is required that minimises data forwarding delay between the tiers of the network hierarchy and maximise the number of sensors communicating simultaneously.

*Scalability Requirement.* The number of sensors and their location is fixed throughout the engine test. However, in development and production testing, a small number of sensors may be added later.

*Harsh Environment.* Development and production testing takes place under the extreme environmental conditions summarised in Table 1.

In addition to the extreme vibration and temperature environment within which the wireless communication network has to work, interferences due to other industrial electrical/electronic devices may also have an adverse impact on the performance of the wireless link [3].

*Robustness and Coexistence Requirement.* The communication protocol has to be designed carefully to make the wireless communication robust enough against interference and enable it to coexist with other electrical equipment.

With the recent advances in wireless sensor networks (WSNs), the realisation of low-cost embedded industrial automation systems has become feasible [1, 4]. Small-scale condition monitoring using wireless technologies for engine testing and in-service engine monitoring are well discussed and demonstrated in [2, 5], where the Bluetooth techniques are adopted for small networks. In [6], a Bluetooth-based demonstrator with 5 nodes connecting the thermocouples and sound sensors has been developed for acquisition and visualisation of the engine's temperature. A wireless sensor network for monitoring the health of aircraft engines is described in [7, 8]. In [9] the authors provide an overview of the architectures of wireless networks for engine and aircraft health monitoring.

It has been shown that the MAC protocol dominates the network performance and, recently, many researchers have been engaged in developing schemes that address the unique challenges of industrial wireless sensor networks. A number of MAC protocols have been proposed for wireless sensor networks. The most common MAC is contention-based channel access, namely, Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), in which nodes transmit data if the medium is sensed idle and use a back-off mechanism in case of busy channel or collisions. Both IEEE 802.11 and the IEEE 802.15.4 are based on the CSMA/CA. However, CSMA/CA is not optimal to handle real-time applications with high data rates, various priority levels, and Quality of Service (QoS) requirements. As congestion increases, contention-based MACs spent most time on back-off to avoid collision and the bandwidth and energy are wasted. Although the IEEE 802.11 standard defines a centralized polling-based channel access method, the Point Coordination Function (PCF), to support time-bounded services, this contention-free approach is based on the contention-based DCF (Distributed Coordination Function) and thus is not efficient due to inefficient polling and a large overhead.

In contrast to the distributed contention-based MAC, centralised channel access can avoid collisions and reduce the amount of time used for backing off, making it more appropriate for a real-time high data throughput applications. Synchronous MAC protocols based on Time Division Multiple Access (TDMA) have attracted considerable interest because of their collision-free operation, higher spectrum efficiency, and low power consumption. While the medium access is coordinated by a controlled schedule, the collisions are avoided and the node's duty cycle can be optimised so that sensor nodes may place themselves in sleep mode for a longer time without sensing the medium. A TDMA-like protocol (called MaCARI) was proposed for industrial wireless sensor networks in OCARI project [10]. A polling-based TDMA MAC protocol with duty cycles is proposed for industrial applications and its performance is analysed in [11]. It has been shown that the polling-based MAC protocol is a special case of TDMA and shows a better performance in terms of scalability and self-organization.

## 3. Network Design

Considering the features described in Section 2, a hybrid wireless/wired data gathering architecture is considered suited for high data rate engine testing. Figure 2 illustrates a typical configuration of engine testing, where the whole engine measurement system is comprised of a number of modules with each having its own communication system to transmit the data collected within it. It is natural to divide

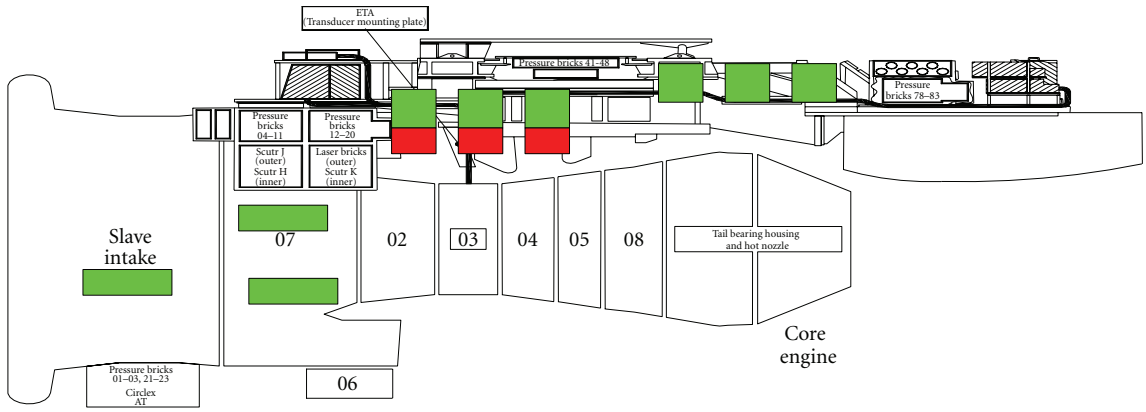



Figure 2: Typical engineering configuration, with permission of Rolls-Royce.

the sensors into subsets allocated at different engine modules and, in turn, the whole communication network is comprised of a set of linear cluster networks, which is ideal for providing communications in systems that have relay stations deployed along a line. As a result, the wireless sensors are hierarchically organised into clusters using the tree-cluster architecture which has been adopted in the recent standard specifications (e.g., the 802.15.4 standard [12] and the ZigBee Alliance specifications [13]).

As shown in Figure 3, the proposed linear cluster network (LCN) consists of three kinds of devices, *Sensor Nodes (SNs)*, *Cluster Heads (CHs)*, and *Wired Sinks (WSs)*, and they are hierarchically organised into clusters. A sensor node connects to a set of transducers (channels) and samples the physical signals periodically. A group of SNs which are close to each other geographically or related in terms of physical measurements comprises a cluster. Each cluster has a special node as its own CH and the SNs transmit their data to the associated CH via a single-hop communication link. Usually, the CH would have more computation capability and storage space than the SNs and a SN may be associated with one more cluster to improve the reliability and scalability. Furthermore, one cluster may be too far to communicate directly with the wired sink (WS) by single hop. In this case, the CHs relay the data received from the child SNs to the WS, either directly or via a multihop path through other intermediate CHs.

In the proposed LCN topology, these clusters and their CHs are organised in a chain with each chain being deployed along the surface of the engine module. As shown in Figure 3, the whole LCN comprises of multiple lines of linear clusters in parallel and each chain, referred to as line (i.e., line 1, line 2,...), consists of a set of cluster heads and a WS. The linear cluster structure is a combination of star and mesh topology. In such a convergecast network there are two kinds of communication, namely, single-hop SN-to-CH communication and multihop CH-to-CH communication. From the proposed engine testing application perspective, in order to achieve a higher network throughput, it is reasonable to make use of all available radio channels, while we assume that the CH nodes are equipped with a double-radio wireless module. The double-radio module potentially uses two different standards and different frequency bands to avoid cochannel interference between SN-to-CH and CH-to-CH communications. This independency enables us to simulate and study the behaviour of the sensor-to-CH and CH-to-CH communication independently. In this paper, we focus on the SN-to-CH communication and study the performances of CSMA/CA and polling in SN-to-CH. With some minor modification, these protocols can be applied to CH-to-CH communication.

Another benefit of the linear cluster network is the simplification of routing protocols, which helps to reduce the communication overhead, save CPU time and energy consumption, and improve the robustness and life time. As the SN always sends the data to its associated CH in a single-hop manner, there is no need for routing. The multihop routing functions are needed only within the cluster heads. Since the LCN has a linear topology, the routing protocol is simplified to a great extent. It is worth noting that, depending on the transmission power, the interference range of CHs may cover the whole network. Thus, the routing problem turns into a media access problem and is solved by a joint design of MAC-routing protocols. The performance of the routing protocol in CH-to-CH communication depends on the underlying MAC protocols which is the focus of WIDAGATE.

## 4. Modelling the Radio Environment

A prerequisite to the engineering of WSN in any environment is a physical-layer wireless channel model that can be used to predict the channel characteristics. In the context of WIDAGATE this implies the channel between any pair of nodes lying on a gas turbine engine surface. Such models may be narrowband or broadband. The former is simpler and appropriate if the dispersion of the channel is small compared to the symbol duration of the signals which the channel will carry. The latter is more complex but must be used if dispersion is a significant fraction of symbol duration. The geometry of a gas turbine engine is essentially cylindrical and an estimate of the maximum data rate that can be properly accommodated by a narrowband channel model (i.e., flat fading without equalisation) can be made by considering

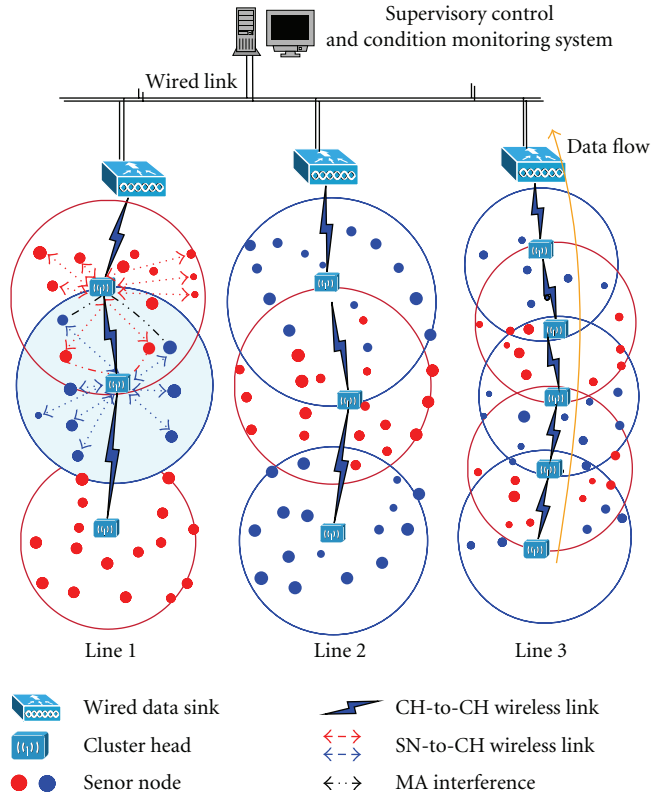



Figure 3: Linear cluster-tree topology (linear cluster network).

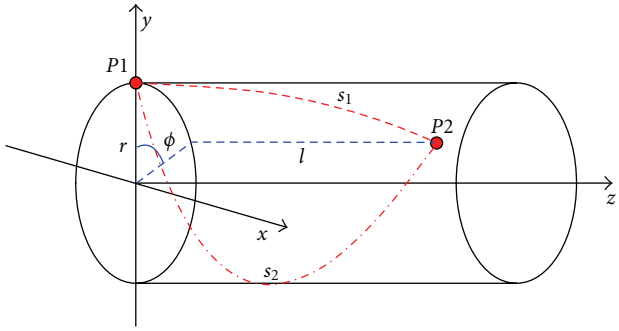



Figure 4: Schematic diagram of two-path geometry.

two-path propagation between a pair of nodes as shown in Figure 4.

The maximum differential path length for any pair of nodes is given by

$$\Delta s = s_2 - s_1 = \sqrt{[r(2\pi - \phi)]^2 + l^2} - \sqrt{(r\phi)^2 + l^2}, \quad (1)$$

where $r$ is engine radius, $\phi$ is angular separation between $P_1$ and $P_2$ projected onto a plane perpendicular to the engine axis, and $l$ is the separation of the planes perpendicular to the engine axis containing $P_1$ and $P_2$. And the corresponding differential propagation delay for a propagation velocity is $\Delta T = (1/c)(\sqrt{l^2 + (2\pi r)^2} - l)$. Assuming (worst case) binary modulation and that time dispersion must be not greater than 10% of the symbol duration, the maximum bit-rate is given by

$$R_b \leq \frac{0.1}{\Delta T} = \frac{0.1c}{\sqrt{[r(2\pi - \phi)]^2 + l^2} - \sqrt{(r\phi)^2 + l^2}}. \quad (2)$$

Choosing extreme values of $\phi = 0$, $r = 1.0$ m, and $l = 2.0$ m then $R_b$ is limited to 6.53 Mbit/s. Since this is greater than the bit-rate envisaged from each SN in the WIDAGATE application a narrowband channel model is appropriate. This suggests that the multipath in engine testing environment has negligible influence on the link quality and thus there is no need for a multipath model. Nevertheless, besides the path loss, the thermal noise and environmental interference are also taken into account in our model (as shown later in this section) to give a better link quality simulation [3].

*Channel Measurements.* The scattering transmission parameter $S_{21}$ was measured across the ISM frequency band (2.4–2.5 GHz) between pairs of points distributed over a rectangular grid on the cylindrical surface of a Gnome gas turbine engine. This particular frequency band was selected based on the majority of WSN devices currently available. The measurements were made between a pair of low-gain (approximately 0 dB) omnidirectional microstrip antennas using an Agilent N5230A vector network analyser both in the absence of (dataset 1) and in the presence of (dataset 2) an engine cowling. The arrangements of measurement grid points for the two datasets are shown schematically in Figure 5.

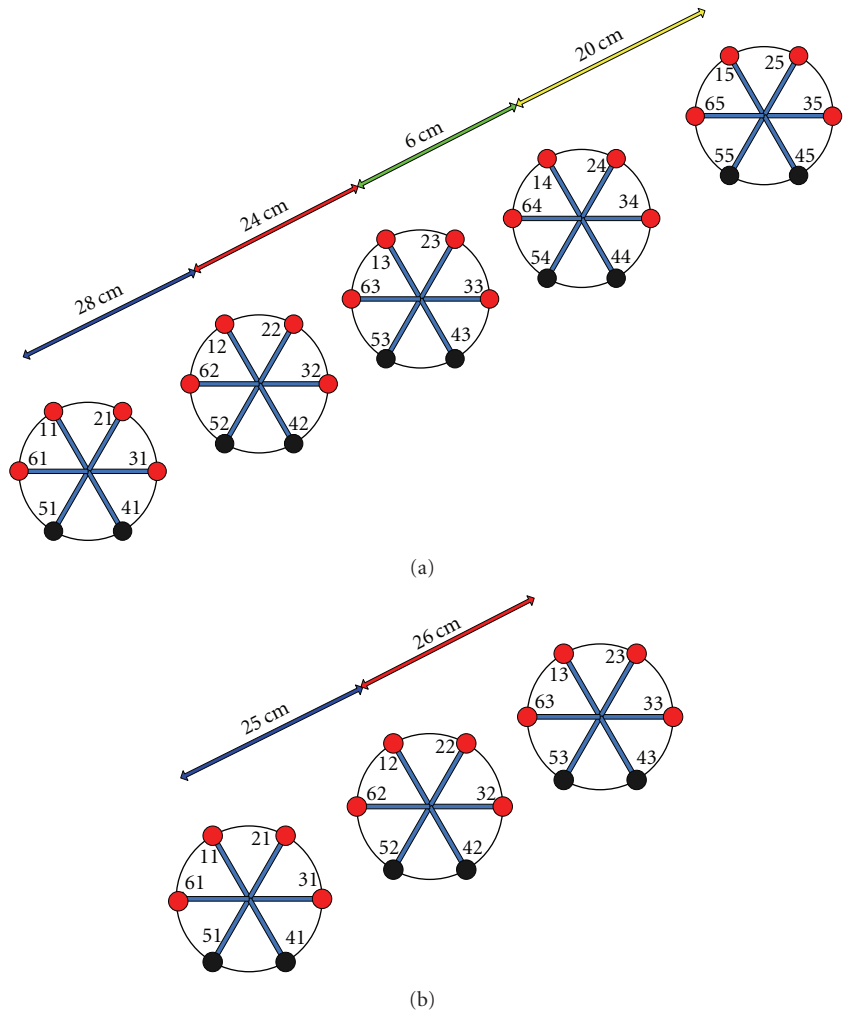



Figure 5: Schematic diagram of engine measurement points: (a) without cowling: dataset 1 and (b) with cowling: dataset 2. Black (darker) points omitted.

Dataset 1 was obtained in the Gnome Test Laboratory at Rolls-Royce in Derby, UK. The engine for these measurements had no cowling (Figure 5(a)). Six potential measurement points were distributed evenly around the circumference of the engine in five planes perpendicular to the engine axis along the length of the engine. The separation between adjacent planes was 28 cm, 24 cm, 6 cm, and 20 cm. There were, therefore, 30 potential measurement points in total. Since measurement time was limited, and since measurements between all pairs of potential points include redundant geometries, two points on each plane were omitted from the measurement process. These are represented by hollow circles in Figure 5. The total number of measurements made was 136. The residual redundancy in measurement geometries, however, means that all geometries are satisfactorily represented in the measurement database.

Dataset 2 was obtained in the Radio Science and Wireless Communications Laboratory at the University of Strathclyde, UK. The Gnome engine used was identical in type but a different instance to that used for dataset 1. The surface detail of the two engines was similar but not identical. Since the surface detail represents an essentially random distribution of scatterers the use of two engines is not thought to materially reduce the usefulness of the resulting statistical model. These measurements were made in the presence of an engine cowling manufactured by SCITEK Consultants Ltd from stainless steel mesh to a specification provided by Rolls-Royce. Access to the measurement points was via slots cut in the cowling. When not being used the slots were covered by aluminium foil fixed in place using a conducting grease. We selected an Al foil with a typical thickness of 16 micron which is 16/1.66 = 9.64 skin depths. The RF attenuation provided by the foil is then $20\log_{10}e^{9.64} = 84$ dB. The protection against leakage (out of one slot and back in another slot) is therefore 168 dB. The RF leakage is therefore considered negligible.

*Channel Modelling.* An empirical transmission loss model has been derived from the measurement datasets. In order to make the model generic, such that it can be applied to engines of arbitrary size, the model is parameterised in terms of path length, $s$, and path curvature, $\kappa$. The path length of

Table 2: Transmission model coefficients.

| | Coefficients | | | |
|---|---|---|---|---|
| | A | B | C | D |
| Dataset 1 | −0.99986 | 0.00043 | −0.01678 | −0.30833 |
| Dataset 2 | −0.99983 | −0.010042 | −0.01532 | 0.02659 |

each measurement is that of a helical segment connecting transmitter location ($P_1$) and receiver location ($P_2$). The arc length $s$ is given by

$$s = \sqrt{(r\phi)^2 + l^2}. \tag{3}$$

The path curvature, reciprocal of radius of curvature, is given by

$$\kappa = \frac{r}{r^2 + (l/\phi)^2}. \tag{4}$$

The mean transmission gain in dB (<0), $\overline{G_T}$, has been modelled as a function of $s$ and $\kappa$ using

$$As + B\kappa + C\overline{G_T} + D = 0. \tag{5}$$

The resulting best-fit surface for datasets 1 and 2 is shown in Figure 6. Each transmission loss measurement in Figure 6 is the average of 6401, equally spaced, spot frequencies within the 100 MHz ISM band. Table 2 contains the best-fit coefficients for each dataset.

Since the scatter of points about the model is large (due to the large random variation of the engine surface from that of a smooth cylinder), an error model for the quantity

$$\Delta = G_{T,\text{measured}} - \overline{G_T} \text{ dB} \tag{6}$$

has also been derived by quantising the 2-dimensional space spanned by $s$ and $\kappa$ into a 4 ($s$) by 3 ($\kappa$) grid and calculating the mean and standard deviation of the resulting histograms of $\Delta$ within each 2-dimensional quantisation interval. The dependence of $\mu$ on $s$ and $\kappa$ and of $\sigma$ on $s$ and $\kappa$ is then found using the same approach as that used for $\overline{G_T}$. The final value of transmission gain thus becomes a sum of a deterministic and a random component, that is,

$$G_T(s,\kappa) = \overline{G_T}(s,\kappa) + \Delta[\mu(s,\kappa), \sigma(s,\kappa)]. \tag{7}$$

The measurements and modelling described above were specific to the ISM frequency band. Further measurements have been made to extend the frequency range of the model up to 11 GHz and a source of thermal noise (determined by the receiving sensor node noise bandwidth, noise temperature and antenna temperature) has been incorporated. Should a bit-by-bit simulation be necessary a time-series model of interference drawn from the standard EUROCAE ED-14E [14] has also been made available. (The simulations presented here are packet level only and replace interference with an equal amount of white Gaussian noise.) Figure 7 is a block diagram of the complete channel model. This channel model has been implemented using Simulink.

## 5. MAC Protocols for Operation

In the proposed LCN topology, the data transmission involves two main communication schemes: the single-hop SN-to-CN communication and the multihop CN-to-CN communication. Since the SN-to-SN is a single-hop communication, its performance is dominated by the medium access control scheme. Although the CSMA/CA protocol has been widely accepted in wireless communication, considering the requirement of WIDAGATE that demands high data throughput, a predesigned, demand-based, and scheduled bandwidth allocation scheme with higher spectral efficiency is more favourable. In this section, a polling protocol is proposed and its performance is compared with the CSMA/CA. The simulation results of both CSMA and polling MAC protocols for SN-to-CH communication are presented, being of greatest concern for the operation of the network. For the CH-to-CH communication, due to the simplified topology of the proposed LCN, the routing protocol among CHs turns into a linear routing scheme and the CH-to-CH multihop communication is dominated by the MAC protocols as well. Hence, the results and conclusions achieved for SN-to-CH can be extended to the CH-to-CH with some minor modification.

*5.1. Configuration and Performance Metrics.* For a fair comparison of different MAC protocols, the network configuration (packet length, etc.) and performance metrics are described as follows. According to the engine testing requirement, all sensor nodes generate 102 Byte (40 Bytes payload for 8-channel measurement data plus overhead like channel ID, time stamp, packet header, etc.) data packets periodically at a sampling interval of 0.03 s. Both the acknowledgement packet (ACK) and the polling packet (REQ) have the same length fixed at 38 Bytes (14 Bytes for MAC layer and 24 Bytes for long preamble PHY layer). For the purpose of real-time data transmission in engine testing, the throughput is calculated on the basis of how many DATA packets are received during the course of an engine test run. Note that, since all the data are buffered at the SNs, some data may be transmitted to the data sink after the end of engine testing. Let $\text{Thr}_{\text{PKT}}$ denote the number of received data packets while engine is running, an effective data bandwidth (throughput), $\text{Thr}_{\text{EDB}}$, in bits per second (bps) is defined as

$$\text{Thr}_{\text{EDB}} = \frac{8rL}{T}\text{Thr}_{\text{PKT}} \text{ (bps)}, \tag{8}$$

where $L$ is the packet length (in Byte), $r$ is the payload-overhead ratio of DATA packet, and $T$ is the duration of engine testing. The latency, termed as sampling-to-receiving delay (SRD), is measured as the time from the SN sampling the physical signal to the data packet being received by the CH. The SRD includes queuing delays at the MAC layer and usually is longer than the access delay. If retransmission occurs due to transmission failure, a large SRD may appear which might be greater than the sampling interval. If failed packets are not discarded, then the SRD will accumulate.

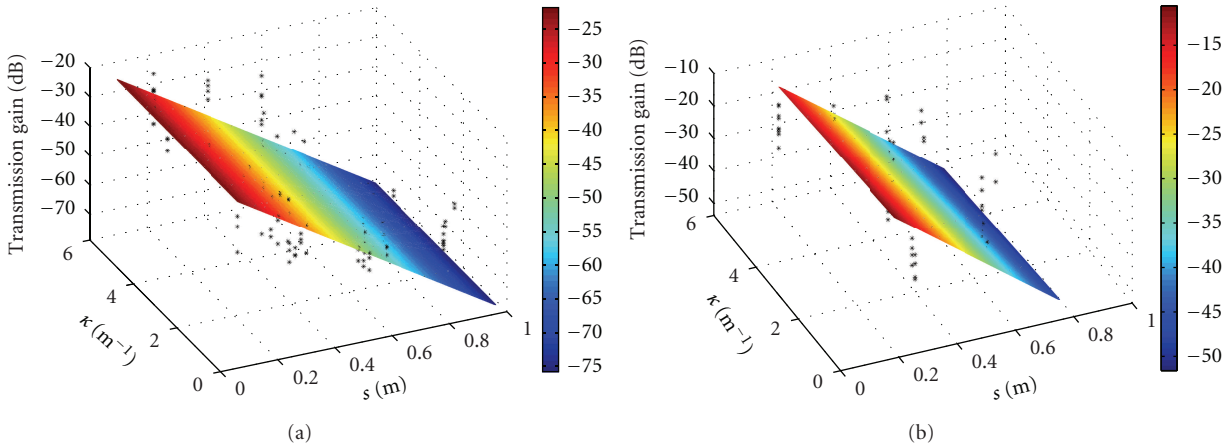



Figure 6: Best-fit plane surfaces to measurements (a) without cowling and (b) with cowling.

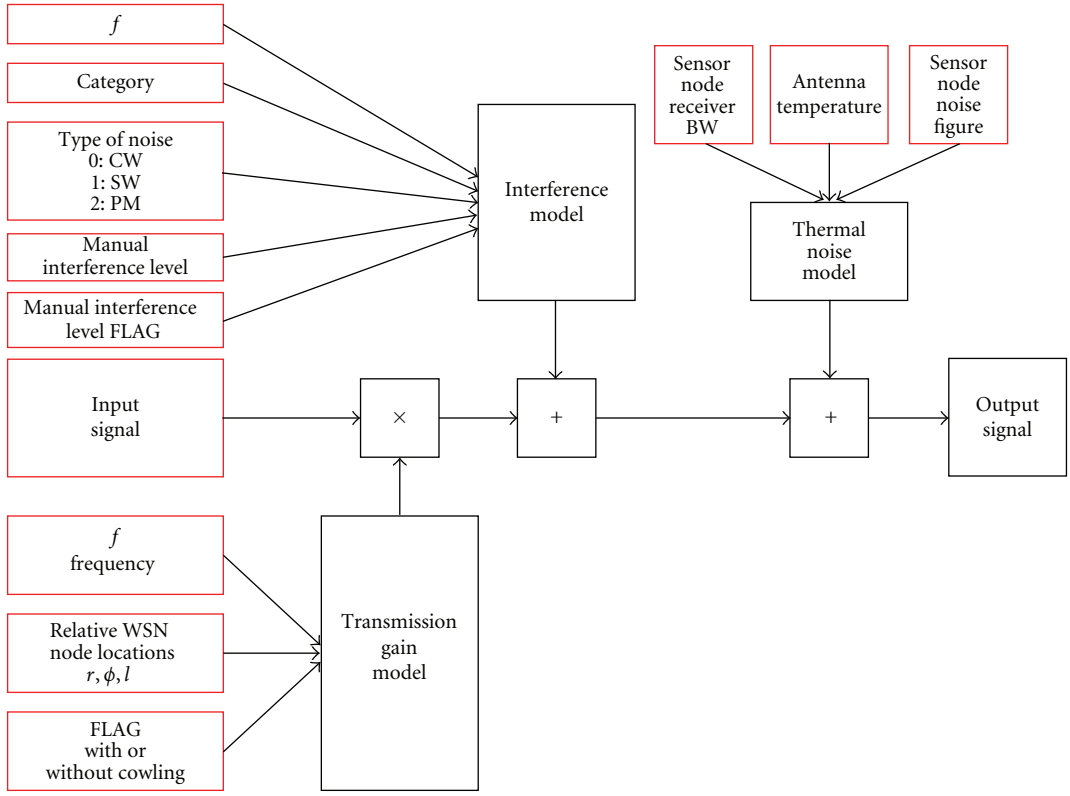



Figure 7: Channel model.

### 5.2. MAC Protocol Description

*5.2.1. CSMA/CA Protocol.* CSMA/CA is a decentralised random access mechanism in which nodes decide autonomously when a packet transmission starts. A node wishing to transmit must first sense the radio channel to determine if another node is transmitting. If the medium is not busy, the transmission may proceed. The CSMA/CA protocol avoids collisions by utilising a random back-off time if the node's physical or logical sensing mechanism indicates a busy medium. The data delivery in CSMA/CA is based on an asynchronous, best-effort, connectionless delivery of MAC layer data with no guarantee that the packet will be delivered successfully. More details of the CSMA/CA can be found at [12].

*5.2.2. Polling Protocol.* The polling MAC protocol is a centralised access mechanism. Although it works by a "listen before talk scheme," the SNs in a polling protocol listen to the request packet from the CH rather than by carrier sensing.

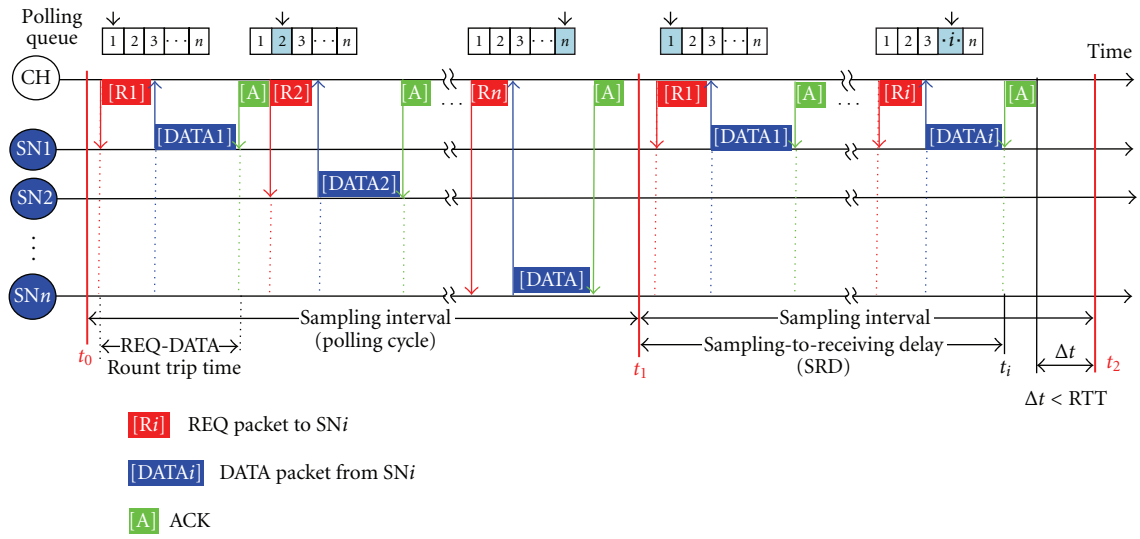



Figure 8: Illustration of the polling MAC protocol.

As shown in Figure 8, the operation of the proposed polling MAC protocol can be described in terms of cycles. Each cycle (which is the same length as the sampling interval) starts at the beginning of a sampling period by polling the child sensor nodes one by one in the cluster. At the SN side, once a measurement is sampled, a DATA packet is generated and queued at the MAC layer to wait for the polling packet from the CH. At the CH side, the CH's MAC layer maintains a polling queue storing the SN's IDs (i.e., MAC addresses). When a new polling cycle starts, the CH reads the first element of the polling queue and broadcasts a data request packet REQ. Only the SN matching the ID accesses the medium by replying with its DATA, while all other SNs keep silence. Once sending out a REQ, the CH sets a timeout. If no DATA packet is received within this timeout, the CH will poll the next SN. The selection of the timeout value depends on the REQ-DATA round trip time (RTT). The RTT in our scenario is 752 $\mu$s for a 102 Bytes data packet at 2 Mbps data rate. If the CH receives the required data packet within the timeout, it replies with an ACK, followed by retrieving the next ID from the polling queue and sending a new REQ to poll the next SN. This process is repeated until either the end of the polling queue is reached or a new sampling period starts. For instance, Figure 8 shows two polling cycles for a cluster with $n$ SNs. As the polling queue is an increasing sequence from 1 to $n$, SN1 is always polled first followed by SN2, 3 and so on. The polling in the first cycle ends when all the $n$ sensor nodes are polled and none of them have any data to send. The second polling cycle ends earlier at $i$th polling ($i < n$), because the time $\Delta t$ left for polling a node before next sampling is less than a REQ-DATA RTT. The reason for terminating the polling cycle earlier is to reduce the SRD and avoid the delay accumulation, because new sampled data will be ready for transmission when the new sampling period starts.

Note that the polling sequence determines when an SN will be polled, thus the medium access is coordinated by the CH and the collision is avoided. The polling queue can be configured as a random sequence or, by default, an increasing sequence as shown in Figure 8. Hence, the bandwidth allocation is fully controlled by the CH and can be adaptive according to the data's' priority and SRD requirement (e.g., for safety critical data) or be random indicating that all sensor nodes have the same priority. It is also worth noting that the polling protocol does not require time synchronisation to avoid collision. In order to guarantee every data packet is received by the CH, the proposed polling assumes a no packet drop policy, which means a DATA packet will be kept at the SN unless an ACK is received. This is implemented by a DATA queue at the SN's MAC layer.

*5.3. Performance Comparison.* Turning our attention to the simulation results of the IEEE 802.11 CSMA/CA and the proposed polling MAC protocol, the network throughput, packet loss, and sampling-to-receiving delay (SRD) are simulated and depicted in Figures 9 and 10.

*Throughput.* Figure 9 shows the relationship between the throughput and the cluster size, where the cluster size is defined as the number of child SNs in the cluster. Values of throughput are shown in terms of both how many packets are received by the cluster head in 30 seconds and the effective throughput (kbps). Two groups of throughput performances are shown in Figure 9. One is the throughput in an interference-free environment, while the other is subject to external interferences (EIs). The throughput achieved in the interference-free environment can be regarded as the upper bound of the throughput, since the throughput purely depends on the protocol itself.

In the interference-free case, as the cluster size increases from 5 to 26, the traffic load increases from 53 kbps to 275 kbps, and the throughput of CSMA/CA goes up steady and linearly with respect to the increasing traffic load. However, when the cluster size is greater than 26, throughput goes down steadily after reaching the maximum throughput of

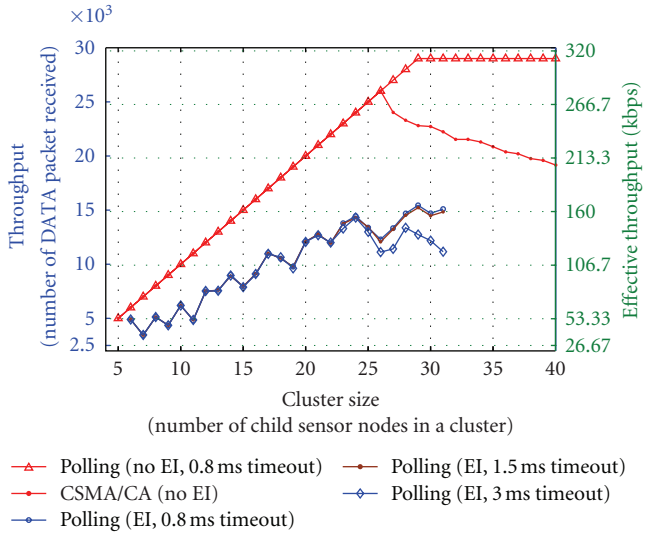



Figure 9: Network throughputs. Throughput of CSMA/CA and polling.

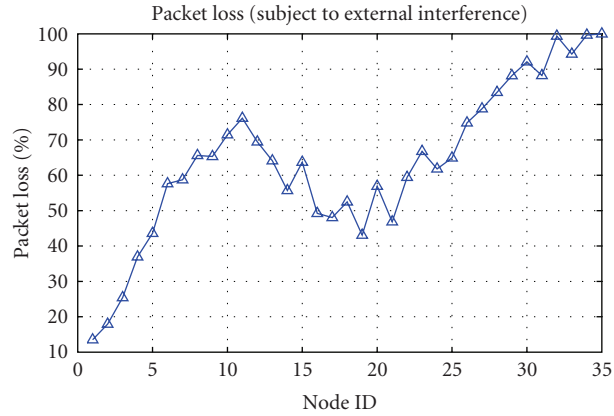



Figure 10: Packet loss of the polling protocol subject to the environment interference.

275 kbps. This shows that the CSMA/CA reaches the saturated condition at 275 kbps, representing the maximum network throughput in overload conditions. In contrast, the throughput of the polling protocol keeps increasing linearly until the cluster size is 29 and then becomes fixed at the highest throughput of 310 kbps even when the cluster size increases further. The simulation results also reveal that the polling protocol works better than the CSMA/CA at high traffic loads (i.e., over saturation). This is because, when the network is saturated, more collisions occur in CSMA/CA and more bandwidth is wasted, thus the throughput of the CSMA/CA decreases. Since the polling protocol is a collision-free scheme, no bandwidth is wasted on collisions and a higher throughput is achieved resulting in higher spectrum efficiency. Furthermore, as the polling has a flat throughput at the saturated condition, the polling protocol is more robust than CSMA/CA.

It is worth noting that, although no collision occurs in the polling protocol, the bandwidth has to be split between the transmission of DATA packet and REQ/ACK packet. The polling protocol can be further improved by either increasing the length of data packets (to increase the share of data transmission) or using a multipolling/multi-ACK scheme (to reduce the share of REQ-ACK).

When the external interference is presented, the packet loss is not only due to collisions but also due to the high level of interference. As a result, the throughput becomes lower than the upper bound. It can be seen that the throughput, in the presence of interference, is about half of the upper bound, suggesting packet loss is about 50% when interference is presented. As the REQ-DATA timeout impacts on the network performance, which is particularly true for short DATA packets at saturated condition, Figure 9 shows the throughput of the polling protocol at three timeout values of $\{0.8, 1.5, 3\}$ ms. It can be seen that they are the same at low traffic load whereas a shorter timeout gives a bit higher throughput at high traffic load. This is because, when packet loss occurs, a shorter timeout allows more slots for REQ-DATA exchanges in a polling cycle and more polling can be applied to compensate for the packet loss.

*Packet Loss.* In order to further study the impacts of the SN's location and the polling protocol on the throughput in a harsh environment subject to external interference, the packet loss rates of individual SNs in the polling protocol are depicted in Figure 10. In this simulation, 34 SNs in total are deployed along a belt circling the engine's surface in the clockwise direction. The CH is at the top of the engine, the first node SN(1) is closest to the CH, SN(1)–SN(9) are deployed from top to bottom at the left side of the engine, the SN(10) at the bottom is opposite to the CH (non-line-of-sight), and SN(11)–SN(34) are placed at the right side of the engine from bottom to top towards the CH.

It can be seen that the packet loss rates are affected by nodes' placement at the engine surface and the polling sequence. The first part of the packet loss curve (Node ID = 1,..., 10) is dominated by the SNs position, where all these nodes can be polled in each polling cycle and the node's position at the engine surface is the main reason for the increase of packet loss. Since SN(1) is the nearest one to the cluster head, it has the lowest path loss and thus the lowest packet loss rate. Since SN(10) at the bottom is not line-of-sight to the CH, it has the largest pass loss and its packet loss reaches a peak of 75%. While the node ID increases, the node gets closer to the CH and the packet loss decreases accordingly due to the decreasing path loss. However, when the node ID becomes greater than 15, the polling sequence dominates the packet loss. These nodes are at the tail of the polling sequence having less chance to be polled and the number of data transmitted to the CH decreases. Therefore, the polling scheme at the tail works as a nonuniform polling. The average data loss rate for the first ten nodes (SN($i$), $i = 1, \ldots, 10$) is 45.60%. As the first ten nodes are always polled, the packet loss in the first ten nodes is mainly due to the external interference.

*Sampling-to-Receiving Delay (SRD).* The SRDs of every node and their standard deviation are shown in Figure 11. The

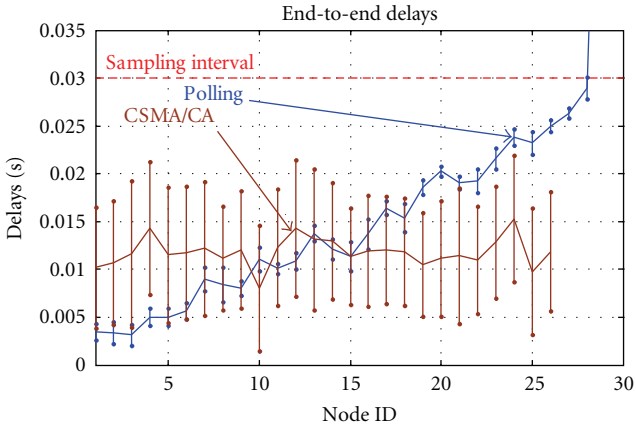



Figure 11: Sampling-to-receiving packet delays in the CSMA/CA and the polling MAC protocols.

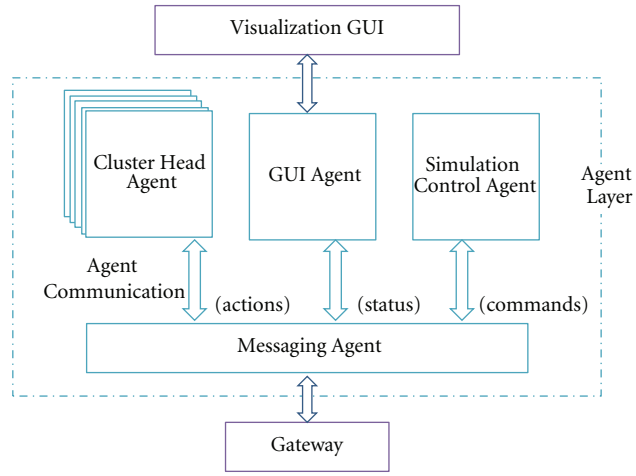



Figure 12: Architectural design of Agent Layer.

blue line is for the polling protocol when the cluster size is 30 and the brown line is for the CSMA/CA of 26 sensor nodes. Due to its random access behaviour, the SRD of CSMA/CA is a random process with a mean of 0.012 ms and an average standard deviation of approximately 0.005. It is noted that, due to the large variances, the maximum SRD of CSMA/CA reaches nearly the sampling interval of 0.03 s. On the other hand, as the polling queue in the polling protocol is a fixed increasing sequence, the smaller ID number the SN has, the earlier the SN is polled. Thus the SRD increases linearly with respect to the SN's ID. Since the SN is polled nearly at the same time slot in every polling cycle, the variance of the SDRs is small with an average standard deviation of 0.0012.

From the simulation results, some conclusions can be drawn. (a) The throughput performance of both CSMA/CA and polling is similar at low and moderate traffic load when the cluster size is smaller than 25. (b) When the traffic load increases further, the network goes into saturation and polling is superior to CSMA/CA, where the throughput of CSMA/CA degrades significantly, but that of polling increases further and supports up to 29 sensor nodes for time-bounded data transmission. (c) In terms of latency, the polling shows much smaller jitter resulting in a better phase relationship among data in engine testing. This is a favourable feature for WIDAGATE. Overall, since WIDAGATE has high traffic load which makes the network saturated, the polling is more appropriate for WIDAGATE in terms of both throughput and latency.

## 6. Agent-Based Control and Optimisation

In contrast to the usual approach of running network simulations as a batch process, the Agent-based application layer provides not only flexibility of logging data for offline analysis and visualisation of the data/metric streams while the simulator core is running, but also provides an interface for the user to interact with the WSN simulator core. The user can make online queries and change node parameters (e.g., location, sampling rate, and traffic load) on the fly. The Agent-based approach also enables an intelligent online optimisation to improve the network performance. More specifically, the multiagent system provides the following functionalities to the WIDAGATE.

(a) Performance Measurement—the Agent Layer serves as an intermediate bridge to enable the user to obtain real-time network performance metrics from the WSN simulator core, for individual cluster head nodes, as well as for the entire simulation.

(b) Visualisation—the Agent Layer provides a Graphical User Interface (GUI) to enable the user to interact with the WSN simulator and visualise the node deployment, as well as network performance metrics and status updates.

(c) Interactivity—the Agent Layer enables the user to change parameters in the WSN simulator, such as changing node locations, so as to evaluate the effects on network performance.

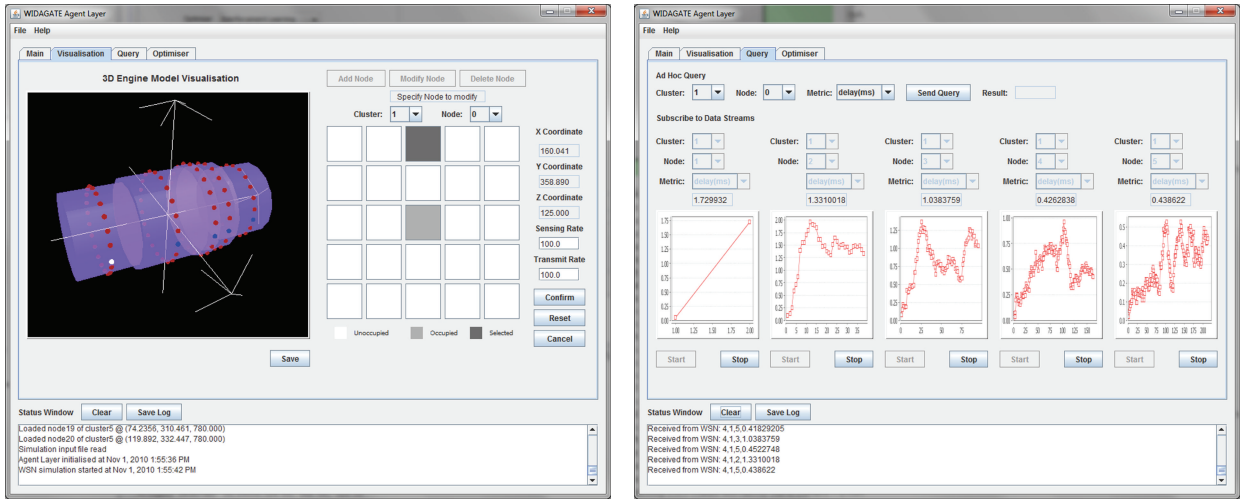
(d) Optimisation—the Agent layer provides an intelligent optimisation algorithm to optimise the location of sensor nodes.

The agent layer architecture and functional design diagram are shown in Figure 12 and the functionalities of each agent module are listed in Table 3. The gateway is at the simulator core providing an interface to the Agent Layer. A TCP socket connection is used for this interface, as this allows the Agent Layer to be abstracted from the WSN simulator. In this way, the Agent Layer module can be easily ported to interface with other underlying platforms, such as a test-bed implementation.

The proposed multiagent system is implemented using JADE (Java Agent Development Environment) [15], as it can be easily ported from development and simulation to a real-world implementation. JADE also provides a set of FIPA compliant (the Foundation for Intelligent Physical Agents (FIPA, http://www.fipa.org/) is an IEEE Computer Society standards organization that promotes agent-based technology and the interoperability of its standards with other

Table 3: Agent module functionalities.

| Agent | Functionality |
|---|---|
| Simulation control agent | (i) Controls the WSN simulator (start/stop/pause/resume) and creates all other agents<br>(ii) Initialises the simulation parameters from input test data and configuration files |
| Cluster head agent | (i) Represents each cluster head in the WSN simulator<br>(ii) Stores cluster head's state information and performance metrics<br>(iii) Implements decision-making and network optimisation |
| GUI agent | (i) Implements 3D visualisation and graph plotting capabilities and interactive GUI for users to control simulation parameters |
| Messaging agent | (i) Implements the agent interface for WSN-Agent Layer integration<br>(ii) Converts ACL (Agent Communication Language) messages in Agent Layer to WSN simulator's message format (e.g., commands, queries) and vice versa (e.g., status updates)<br>(iii) Maintains the socket connection with the WSN Gateway |




(a) Visualisation and node modification tab

(b) Querying and subscription tab

Figure 13: GUI screenshots of the Agent Layer.

technologies) agent messaging protocols for negotiation and decision-making, also known as ACL (Agent Communication Language). The main functions provided by the Agent-based application layer are detailed below.

*6.1. Visualisation.* The developed WSN core simulator is visualised in an online fashion by the multiagent system. The "Visualisation" tab of the GUI (shown in Figure 13(a)) illustrates node locations on a 3D engine model and allows users to adjust the node locations. The "Query" tab (shown in Figure 13(b)) provides capabilities for inputting user requests to monitor specific nodes and online visualisation of the vibration/pressure/temperature data collected by the sensor nodes. The user can make an ad hoc data query or subscribe to a node's performance metric stream, which will be logged for offline data analysis.

*6.2. User Interaction.* The user interaction is implemented in two stages. The interaction between the user and the agent system is done by the GUI agent and the interaction between the agent system and the underlying simulator core is done by the Messaging Agent, which connects to the simulator core through a socket connection for information exchange. The simulator core has a special node, termed as gateway, working as a server and providing a socket port. The gateway is an event scheduler of the simulator core and has full access to all other nodes (i.e., SNs, CHs). Once the simulator core is initialised, the gateway sets up a socket port and listens to the connection request from the agent system. A set of commands are defined for exchanging information between the agent and the gateway (as shown in Table 4).

At the agent side, the Messaging Agent interfaces with the gateway module via a TCP connection (as shown in Figure 14). When an agent module wants to interact with the simulator core, an ACL message is generated and sent to the Messaging Agent. The Messaging Agent interprets the ACL message into the appropriate commands and sends them over the socket connection to the simulator core. These commands are parsed at the gateway and executed by the corresponding modules in the simulator core. In the reverse manner, update messages from the simulator core are collected at the Gateway module and subsequently passed over the socket connection back to the Messaging Agent, which relays these updates to the corresponding Agent modules.

Table 4: Message type definition.

| Value | Msg_Type | Attribute list | Description |
|---|---|---|---|
| 1 | SUB_START | — | Subscribe for gathering performance data from specified node |
| 2 | SUB_STOP | — | Stop subscribing to specified node |
| 3 | QUERY_ONCE | — | Single query of performance measures from specified node |
| 4 | DATA | {performance_measure_list} | List of performance measures from specified node |
| 5 | MOVE_NODE | $x$_coord, $y$_coord, $z$_coord | Move specified node to specified location |
| 6 | ADD_NODE | cluster_id, $x$_coord, $y$_coord, $z$_coord | Add a new node at specified location |
| 7 | DEL_NODE | — | Delete the specified node |
| 8 | ACK_MOVE_NODE | $x$_coord, $y$_coord, $z$_coord | Acknowledge the command to move specified node |

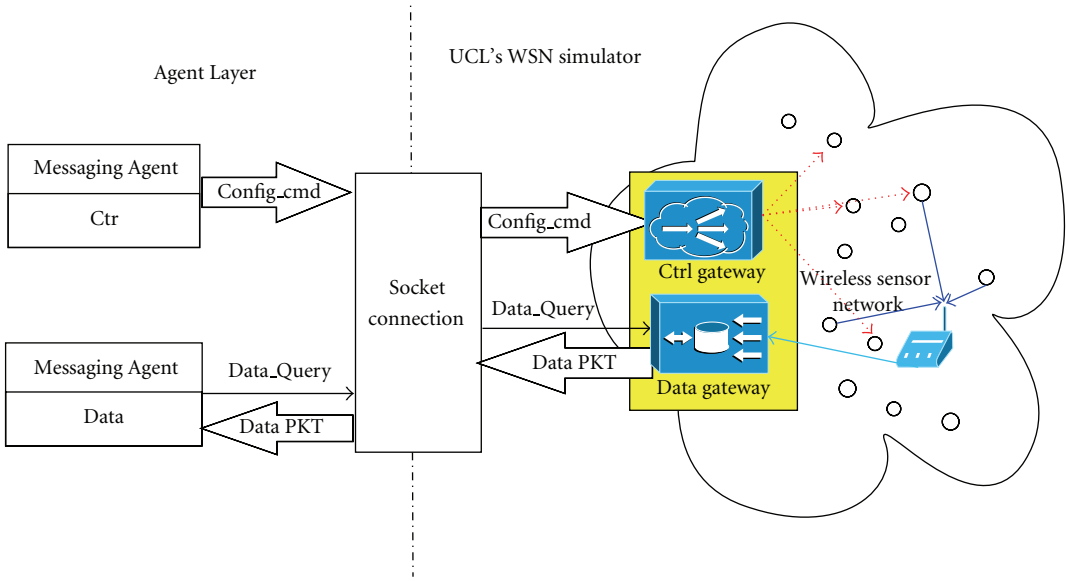



Figure 14: Communication mechanism between Agent Layer and WSN simulator.

*6.3. Multiagent Optimisation.* The ability of Agents to communicate information and make intelligent decisions about meeting objectives has been exploited in this application. In particular, we have incorporated the SN agents with an optimisation capability using which they are able to, as a group of information-sharing cooperative group, determine the most effective topology of the wireless network (with respect to a given performance metric, such as propagation delay).

*6.3.1. Optimisation Formulation.* The SN should be placed within a given distance constraint around the transducer. Given this requirement, the possible locations around a transducer can be formulated as a $m \times n$ grid, as shown in Figure 15, where a $5 \times 5$ grid represents the 25 possible locations for the SN. Choosing one of these grid locations as the new position of an SN is considered an "action" by the corresponding SN agent. Given the node locations (i.e., the $5 \times 5$ grid) is a discrete set of choices, the action space is also discrete. The metric of interest (e.g., packet delivery delay) is nonconvex with respect to the action. As such a discrete nonconvex search method is called upon to determine the optimal locations for the placement of sensor nodes. One such search method is the Reinforcement Learning (RL) approach [16]. Note, the generic RL method searches for optimal "paths" where an agent tries to find the minimum cost (or, maximum utility) traversal from a start to an end location using a metric that is related to the "quality" of each intermediate action or location. However, in our case, we are interested in finding just the optimal location for an SN (with respect to the transducer location) instead of a traversal path. This simplifies our problem formulation.

The search for the optimal location of an SN can be formulated as an iterative RL problem, in which a node keeps track of the *values* of network performance (e.g., throughput, SRD) for each possible action $a \in \{0, 1, 2, \ldots, 24\}$, where each entry corresponds to a square of the 25 grids and the values stored are represented by $Q(a)^2$ (the notation $Q$ is adopted from the standard representation used in RL literature [16]). The parameter to be optimised is the action code $a$ or the SN location. Note this is a simplified RL formulation is a simplified version in that it does not use information about state transition (i.e., the traversal path of the SN).

An action is drawn from the set of all actions using the $\varepsilon$-greedy method [16], such that with probability $\varepsilon$, a node will randomly choose an action from all possible actions, and with probability $(1 - \varepsilon)$, the node will choose the action with

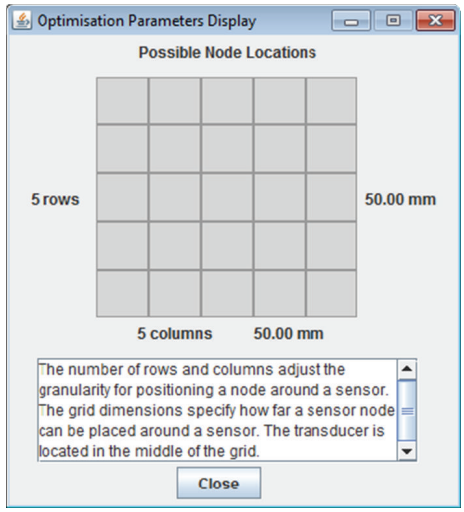



Figure 15: Grid squares where the SN can be placed around a transducer.

the largest $Q$ value (i.e., the *greedy* choice). As the search for the optimal location progresses, the value of $\varepsilon$ is gradually reduced to increase the probability of choosing the *greedy* action, that is, decreasing exploration and increasing exploitation.

*6.3.2. Optimisation Implementation.* The $\varepsilon$-greedy optimisation can be conducted independently by each SN agent (also called SN in the following discussion) to determine their optimal locations within their respective location grids. Another approach is to allow SN agents to share information with their neighbouring agents. When an SN chooses an action (i.e., it selects one of the grid locations), it queries its neighbours to rate the action and an action is chosen according to their past experiences (in this case a neighbour is another SN that is nearby to a given SN which is reachable wirelessly). SNs communicate with their neighbours wirelessly to perform query and receive ratings. A neighbour rates the action based on its effect on the local metric of interest. There could be instances in which an action that is deemed good for a node may be seen as bad by a neighbour node.

In order to reduce the number of iterations, the linear cluster topology is taken into account, where a CH agent performs optimisation for all nodes in its cluster. CH agents interact with other CH agents by exchanging action values. These interactions are shown in Figure 16. This method requires the CH agent to be aware of the states of all sensor nodes within its cluster, which can be supported by the polling protocol.

This design is modular in that the same local optimiser code can be ported to new cluster head nodes easily, which would subsequently interact with the nearest cluster head nodes to carry out the optimisation. The user initiates the optimisation from the user interface (as shown in Figure 17).

At the initial stage, the GUI Agent distributes the optimisation parameters ($m$, $n$, $a$, the objective function and stopping criterion) to each CH Agent and the CH Agent then initialises the action-value and visit-count for each of the $m \times n$ grid squares. The CH Agent then triggers the optimisation process by sending a metric query via the Messaging Agent to the simulator core. While the simulator core is running, the CH Agent collects the metric of interest from the messages returned by the gateway. The average across a few collected data points is computed before a new action is chosen based on the $\varepsilon$-greedy algorithm depending the current action values. The action values are updated accordingly by either the SARSA update rule [16]

$$Q_a \longleftarrow Q_a + \alpha(r + \gamma Q_{a+1} - Q_a) \tag{9}$$

or the $Q$-Learning update rule [16]

$$Q_a \longleftarrow Q_a + \alpha\left(r + \gamma \max_{a+1} Q_{a+1} - - Q_a\right), \tag{10}$$

where $\alpha$ is the learning rate, $r$ represents the immediate reward which, in this case, depends on the objective value (e.g., SRD), $\gamma$ is the discount rate controlling the amount of consideration the optimiser gives to rewards of subsequent actions, and it determines how much the optimiser values future potential gains relative to immediate rewards.

The computation of new action lies in a range of integer values $\{0, 1, 2, 3, \ldots, 24\}$. The integer representation of the chosen action is then converted into the corresponding row and column in the location grid. The row and column values are then converted into the corresponding 3D coordinates, using the transducer locations and the engine model. The new location is then sent to the simulator core via the Messaging Agent and takes effects immediately. The entire process is repeated until the set of locations converges, that is, it does not change across iterations. The solution is then reported to the GUI Agent, which displays the optimised node locations on the GUI. The user can then choose to try out the proposed node locations.

The RL-based optimization algorithm is tested on a simulation where the objective is to determine the location with the minimum value on a $5 \times 5$ grid. We choose propagation delay as the metric of interest in this test study (any other real-valued metric can be used instead). In this simulation, we have set the environment such that one square is set to return a mean value of 1 ms; one square is set to return a mean value of 5 ms; the rest of the squares return a mean value of 10 ms. The standard deviations for all locations are set to be 5 ms. The convergence of the search process is shown in Figure 18, where the $x$-axis shows number of iterations and the $y$-axis the delay (s).

In this simulation, it took about 260 iterations for the search to converge. This is because all alternative locations (the individual squares in the grid) had to be visited a few times to determine the mean value before confirming that a particular location returns the lowest value.

## 7. Validation through an Engine Testing

In the second stage of the development, validation of the simulation environment was carried out by testing the wireless sensor nodes performance on two engine platforms: the

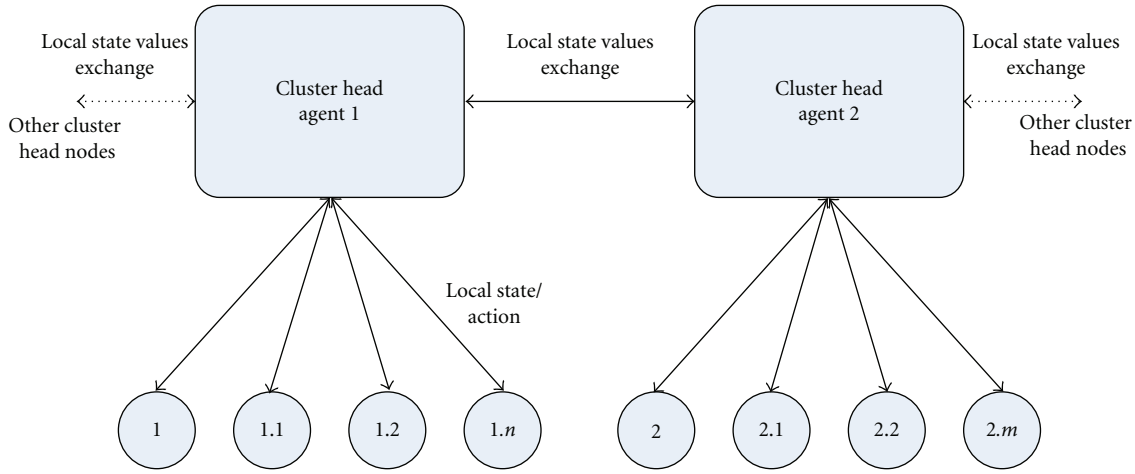



Figure 16: Value exchanges in distributed reinforcement learning-based formulation. Note: *x.0* refers to the cluster head node in cluster *x*, whereas *x.1*,. . .,*x.n* or *x.m* refer to sensor nodes within cluster *x*.

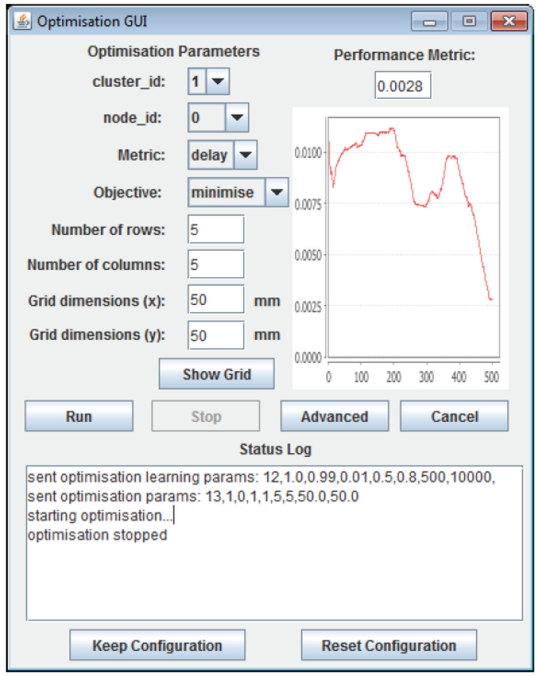



Figure 17: GUI to set parameters and launch optimisation.

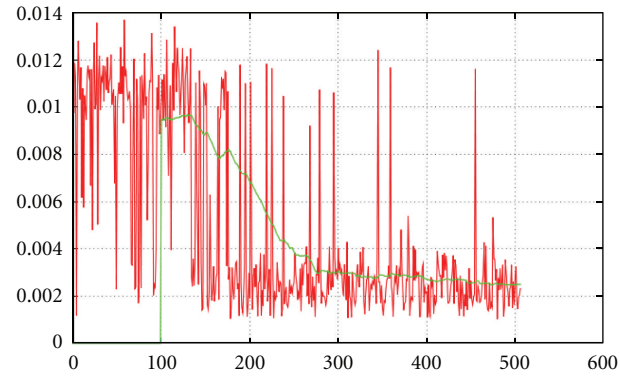



Figure 18: Convergence graph of the optimiser determining the optimal node location.

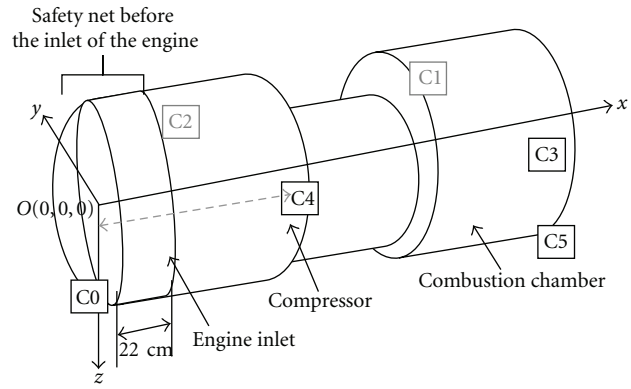



Figure 19: A schematic diagram showing the CISP node locations.

Gnome and the Trent 900. The sensor nodes were the CISPs, provided by SELEX Galileo, UK with the tests carried out in the test facilities of Rolls-Royce, Derby, UK. The main tasks and objectives of the tests were (a) to test the radio model and interference when operating the CISP nodes within an engine; (b) to test the network communication performance in terms of data throughput and packet loss; (c) to validate the simulation model, specifically the transmission gain model.

*7.1. Gnome Engine Test Package Arrangement and Equipment.* The Gnome test setup is shown in Figure 19, where the engine was fixed on the ground with six CISP nodes mounted onto the engine frame via brackets. Their locations are C2 and C4 (on either side of the compressor), C0 (at the engine inlet), C5 (at the back of the combustion chamber), and C1 (to the left of the combustion chamber) with the addition

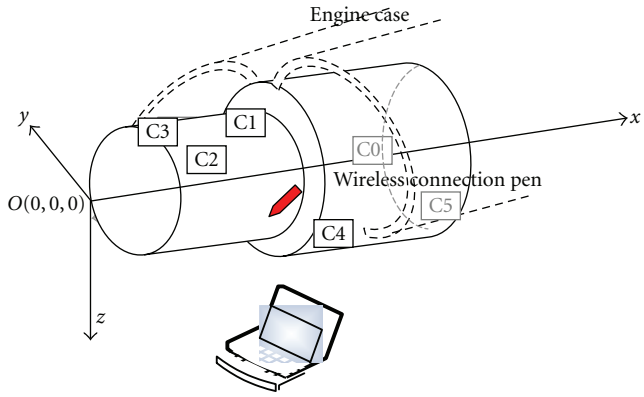



Figure 20: Trent 900 engine test setup and a schematic diagram of CISP node locations.

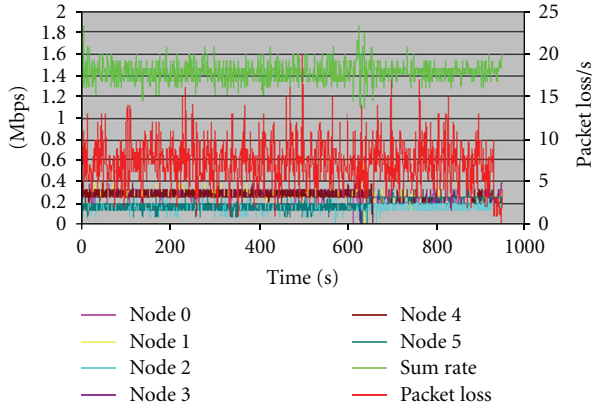



Figure 21: Six CISP nodes data rate and packet losses for static Gnome engine test.

of sensor node 3 which was to the right of the combustion chamber (C3). A schematic diagram of CISP node locations is shown in Figure 19. A single cluster topology was set up to test the CSMA/CA protocol to allow us to validate the simulation platform. As the CISP nodes are commercial hardware adhering to the 802.11 standard other MAC protocols could not be tested. Implemented on the CISP device all the nodes in the test were communicating back to a CH unit located in the control room of the engine test facilities. During the first set of tests, a single communication channel, Channel 6 (2.437 MHz), was used.

*7.2. Trent 900 Engine Test Procedure.* The second set of tests were carried out on the static Trent 900 engine in the Rolls-Royce training centre, Figure 20. The locations of the CISP nodes were C0 (Oil tank on compressor chamber), C1 (Underneath the compressor chamber), C2 (Compressor chamber exit), C3 (Turbine exit), C4 (Located at the gear box level), and C5 (Engine inlet). The CH (a laptop) was located near the external gearbox drive shaft, approximately in line with node C1. A single cluster topology was adopted and a wireless "sniffer" pen placed inside the engine in order to act as the cluster head as it would not have been practical to place the laptop itself in the case of the Trent 900.

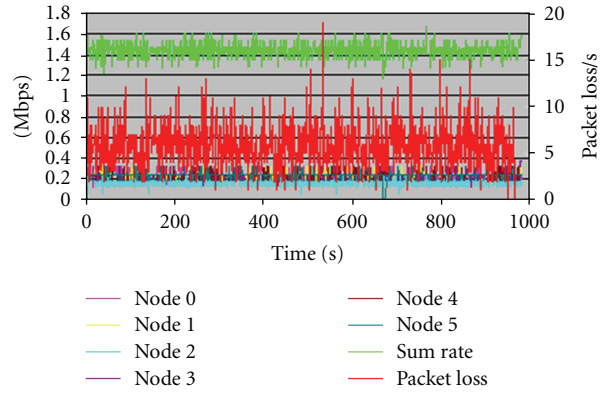



Figure 22: Six CISP nodes data rate and packet losses for running Gnome engine test.

*7.3. Test Results.* The study concentrated on benchmarking the performance of the WSN in terms of communication channels, communication protocols in the two case scenarios of (a) an unpowered and (b) running engine, given the sensor nodes were kept at the same location. The network performance, or quality of service (QoS), is measured by throughput and packet loss. In the next sections an example of the test readings for six CISP nodes on both engine types is given.

*Gnome Engine Tests.* The average data throughput for static and running Gnome engine are 1.435 Mbps and 1.440 Mbps, respectively, while the average packet losses are 7.245 and 5.893 (packet loss per second), respectively. The throughput and packet loss with respect to time are shown in Figure 21 for static engine and Figure 22 for a running engine. In the running test, it is possible to observe that the data throughput is more evenly distributed across the network, with nodes 0, 1, 3, 4, and 5 carrying a load of approximately 0.25 Mbps and node 2 carrying a slightly smaller data rate of approximately 0.17 Mbps.

*Trent 900 Engine Tests.* As shown in Figure 20, six CISP nodes were deployed around the engine core and fan case and covered by the cowling. The throughput and packet loss of the six CISP test are shown in Figure 23. It is possible to observe that again node 1 and node 2 present an analogous behaviour with an approximate data rate of ~0.25 Mbps. Node 0 presents a much degraded performance, with approximately 0.1 Mbps transmission and node 3 seems to follow the behaviour of node 0. Nodes 4 and 5 present the highest data rate transmission in the system.

This phenomenon is related to the MAC protocol employed in 802.11. The IEEE 802.11 employs a Carrier Sensed Multiple Access with Collision Avoidance (CSMA/CA) MAC protocol to manage the access of the media, in this case RF over the air. This protocol is an unmanaged protocol which seeks to avoid collisions (i.e., two radios transmitting at the same time) and resolves the situation by backing off for a random period. This protocol works well with channels that are operated with low link utilisation, lower

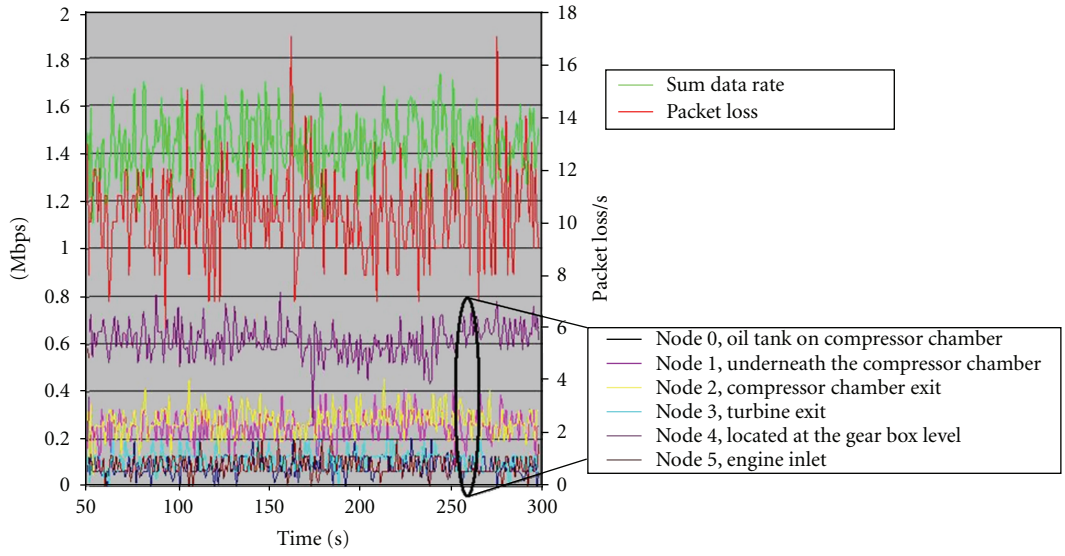



Figure 23: Six CISP nodes data rate and packet losses for Trent 900 engine test.

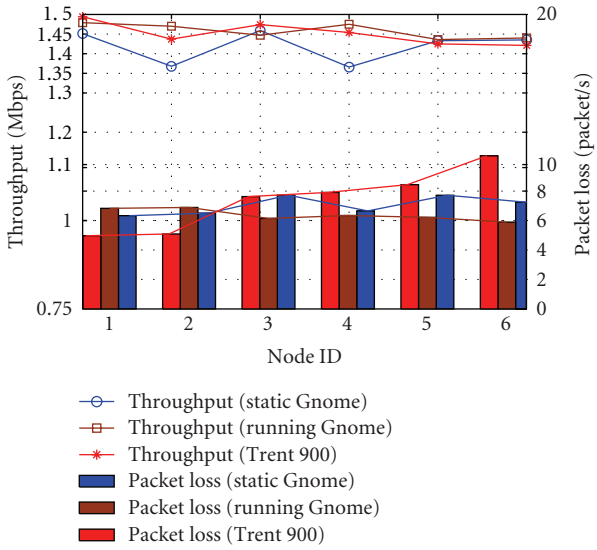



Figure 24: Summary of engine tests results. 6 CISP nodes throughput and packet losses.

than approximately 30%. However, in our engine tests, the wireless network works in a saturation condition where the traffic load injected into the communication network by the sampling process is about 70% of the nominal data rate. It may be that the random back of times in the MAC implementation are actually only pseudorandom, perhaps explaining the cyclic nature of some of the results.

Furthermore, the Trent 900 is a large engine with an irregular shape and surface resulting in the six CISP nodes having different transmission ranges and sensing ranges. It is possible that what we experience during the test is a "hidden node" problem. When the transmitters transmit to the same receiver at approximately the same time, they do not realise, as pointed out in [9], that their transmission collide at the receiver. The hidden node problem in CSMA/CA networks causes unfairness. During the testing, some other RF devices (e.g., Bluetooth) and interference sources may exist and introduce interference to our system. As shown in the literature [17, 18], there is a trade-off between the total throughput and the fairness in CSMA/CA networks with multiple wireless links. The fairness may go worse in saturated networks.

A summary of the data recorded for both the Gnome and the Trent 900 Engine tests is shown in Figure 24 depicting the throughput and packet loss when the number of CISP nodes increasing from 1 to 6. It was possible to observe that as the number of nodes increased there was a slight increase in packet loss rate.

## 8. Conclusions

In this paper, we have described, in detail, the main achievements of the WIDGAGTE project: the development of an accurate and experimentally validated simulation model and a system demonstrator of a wireless sensor network for data gathering and health monitoring during gas turbine engine testing. The deployment of sensors within an aero engine constitutes a harsh environment with complex RF transmission characteristics and therefore a bespoke radio environment model has been developed from experimental data to inform the simulation models. Extensive simulations based on this model have shown that, in such a harsh environment, a polling-based contention-free medium access control scheme with regular duty cycles can outperform contention-based protocols in terms of both the throughput and delays.

An agent-based WSN simulation platform with a user-friendly GUI for customer interaction and optimisation has been developed to control the network simulations to allow an engine test engineer to rapidly develop an optimal network deployment to best suit their testing regime.

The developed software simulation platform and the hardware test bed not only demonstrates the usefulness of wireless technologies, but also helps the end-users build the confidence on the use of wireless technology for engine condition monitoring. The system proposed has a high innovative value, derisking wireless data acquisition in engine testing and potentially allowing in-flight condition monitoring of gas turbine engines, with extension to a wide range of potential aircraft monitoring applications.

## Acknowledgments

This research was supported by the WIDAGATE (Wireless Data Acquisition in Gas Turbine Engine Testing) project sponsored by the Technology Strategy Board (TSB) project Gathering Data in Complex Environments and the UK Engineering and Physical Sciences Research Council (EPSRC) under Grants TS/G002614/1 and TS/G002681/1. The Technology Strategy Board is a business-led executive nondepartmental public body, established by the government. Its role is to promote and support research into, and development and exploitation of, technology and innovation for the benefit of UK business, in order to increase economic growth and improve the quality of life. It is sponsored by the Department for Business Innovation and Skills (BIS). For further information please visit http://www.innovateuk.org/. The authors wish to acknowledge the following collaborative partners and individuals for provision of the engine experimental data and technical support: Graham B. Hesketh, Armin Stranjak (Rolls-Royce plc), Douglas Friedrich, Jason Lepley, Ola Aribisala, David Lloyd, and Graham Bourdon (Selex Galileo, UK).

## References

[1] V. C. Gungor and G. P. Hancke, "Industrial wireless sensor networks: challenges, design principles, and technical approaches," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 10, pp. 4258–4265, 2009.

[2] H. A. Thompson, "Wireless sensor research at the Rolls-Royce Control and Systems University Technology Centre," in *the 1st International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace and Electronic Systems Technology (VITAE '09)*, pp. 571–576, May 2009.

[3] K. Sasloglou, I. A. Glover, P. Dutta, R. Atkinson, I. Andonovic, and G. Whyte, "Empirical modelling and simulation of transmission loss between wireless sensor nodes in gas turbine engines," in *the 7th International Conference on Information, Communications and Signal Processing (ICICS '09)*, December 2009.

[4] L. Krishnamurthy, R. Adler, P. Buonadonna, and J. Chhabra, "Design and deployment of industrial sensor networks: experiences from a semiconductor plant and the North Sea," in *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2005.

[5] H. A. Thompson, "Wireless and Internet communications technologies for monitoring and control," *Control Engineering Practice*, vol. 12, no. 6, pp. 781–791, 2004.

[6] D. Goldsmith, E. Gaura, J. Brusey, J. Shuttleworth, R. Hazelden, and M. Langley, "Wireless sensor networks for aerospace applications—thermal monitoring for a gas turbine engine," in *the NSTI Nanotechnology Conference and Expo (NSTI-Nanotech '09)*, pp. 507–512, May 2009.

[7] H. Bai, M. Atiquzzaman, and D. Lilja, "Wireless sensor network for aircraft health monitoring," in *Proceedings of the 1st International Conference on Broadband Networks (BroadNets '04)*, pp. 748–750, October 2004.

[8] D. M. Benson and H. Bai, "Aircraft engine sensor network using wireless sensor communication modules," US Patent Application 20050213548, 2005.

[9] R. K. Yedavalli and R. K. Belapurkar, "Application of wireless sensor networks to aircraft control and health management systems," *Journal of Control Theory and Applications*, vol. 9, no. 1, pp. 28–33, 2011.

[10] K. A. Agha, G. Chalhoub, A. Guitton et al., "Cross-layering in an industrial wireless sensor network: case study of OCARI," *Journal of Networks*, vol. 4, no. 6, pp. 411–420, 2009.

[11] H. Yang and B. Sikdar, "Performance analysis of polling based TDMA MAC protocols with sleep and wakeup cycles," in *IEEE International Conference on Communications (ICC '07)*, pp. 241–246, June 2007.

[12] IEEE-TG15.4, "Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)," IEEE standard for Information Technology, 2003.

[13] Zigbee-Alliance, "ZigBee specification," 2005, http://www.zigbee.org/.

[14] EUROCAE ED-14E, "A Joint EUROCAE RTCA Achievement," Tech. Rep. Sections 19–21, The European Organisation for Civil Aviation Equipment, March 2005.

[15] G. Caire, *JADE Tutorial: JADE Programming for Beginners*, 2009.

[16] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 1998.

[17] Y. Jian, M. Zhang, and S. Chen, "Achieving MAC-layer fairness in CSMA/CA networks," *IEEE/ACM Transactions on Networking*, vol. 19, no. 5, pp. 1472–1484, 2011.

[18] C. Huang, C. T. Lea, and A. K. S. Wong, "On fairness enhancement for CSMA/CA wireless networks," *IEEE Systems Journal*, vol. 4, no. 4, pp. 511–523, 2010.