TÉCNICO LISBOA

Locality and Interest Awareness for Wireless Sensor Networks (LIASensor)

Bruno Jesus Andrade

Thesis to obtain the master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Prof. Paulo Jorge Pires Ferreira

Prof. Luís Manuel Antunes Veiga

Examination Committee

Chairperson: Prof. João Paulo Marques da Silva

Supervisor: Prof. Paulo Jorge Pires Ferreira

Member of the Committee: Prof. Artur Miguel do Amaral Arsénio

June 2014

Abstract

Nowadays, sensors are becoming increasingly important. The number of equipment and devices using these components are increasing at a high rate; but in reality, it seems that we do not realize that. Thus, the emergence of wireless sensor networks (WSN) was inevitable.

Wireless sensor networks can be described as sets of sensors that are able to gather information, forward it, and exchange it with different nodes.

These modern networks can monitor e control the environment in which we live; but most important, these networks can adapt to several scenarios without significant changes in their topologies.

However, wireless sensor networks have a huge liability: the limited lifetime. Because sensors are powered by limited power sources, it is certain that their power sources will discharge, compromising the entire network.

Regarding this problem, a protocol able to increase the lifetime of wireless sensor networks has been developed. So, I propose **LIASensor** (**L**ocality and **I**nterest **A**wareness for Wireless **Sensor** Networks). A protocol that uses locality and interest awareness with the purpose of reduce the number and size of messages.

To test LIASensor, I developed and executed a system prototype. The prototype assessment was made through the comparison of systems with and without LIASensor. The obtained results were positive and very encouraging. The network with LIASensor decreased both the number and size of messages, which increased the lifetime of the entire network. So, LIASensor proved that there is still much work to do so that the lifetime of wireless sensor networks can increase even more.

Keywords: Wireless Sensor Network , Interest Management , Locality-Awareness , Interest-Awareness

Resumo

Os sensores estão a ganhar uma grande preponderância no nosso dia-a-dia. É cada vez maior o número de equipamentos que utilizam estes componentes, e a verdade é que nem nos apercebemos da sua existência. Assim, tornou-se inevitável o aparecimento de redes de sensores sem fios, isto é, conjuntos de sensores capazes de recolher informação do meio e encaminha-la, sem recurso a um meio físico.

Estas redes modernas conseguem monitorizar o ambiente em que vivemos, e mais importante, conseguem adaptar-se a diversos cenários sem grandes alterações. Contudo, as redes de sensores sem fios têm uma grande vulnerabilidade: o seu tempo de vida limitado. Como os sensores são alimentados por uma bateria limitada, é inevitável que esta se esgote ao fim de algum tempo, comprometendo assim toda a rede.

Esta limitação motivou-me a desenvolver o **LIASensor** (**L**ocality and **I**nterest **A**wareness for Wireless **Sensor** Networks), um protocolo capaz de aumentar o tempo de vida de redes de sensores sem fios.

Para testar o LIASensor, um protótipo do sistema foi desenvolvido e executado, e a sua avaliação foi feita através da comparação de resultados entre redes com e sem o LIASensor. Os resultados obtidos foram positivos e promissores.

O LIASensor conseguiu reduzir tanto o tamanho como o número das mensagens, o que se traduziu num aumento do tempo de vida da rede. O LIASensor prova assim que ainda existe muita coisa que pode ser feita de forma a aumentar o tempo de vida de redes de sensores sem fios.

Palavras-chave: Redes de sensores sem fios , Gestão de interesse , Locality-Awareness , Interest-Awareness

Agradecimentos

Gostaria de agradecer a todos aqueles que me ajudaram ao longo do tempo em que trabalhei na minha tese de mestrado. Para começar, agradeço ao meu orientador, Professor Paulo Ferreira, pela sua orientação, disponibilidade e apoio. Um agradecimento especial a todos os meus amigos e colegas que estiveram sempre disponíveis para me ouvir e apoiar.

Gostaria ainda de agradecer o suporte financeiro da Fundação para a Ciência e Tecnologia (FCT), através da bolsa de investigação.

Um agradecimento especial a todos os meus outros amigos e colegas, que embora não fazendo parte da minha vida académica, estiveram sempre presentes com um espirito de camaradagem e amizade únicos.

Por último, mas não menos importante, gostaria de agradecer à minha família, em especial aos meus pais, pelo seu apoio financeiro e emocional ao longo destes anos de estudo.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Nowadays , sensor nodes are used worldwide in large range of applications. Technology advances [8, 9], have been the primarily responsible for their dissemination around the world, sensors have become powerful, cheap, small and disposable, which allowed the emergence of Wireless Sensor Networks (WSN).

Wireless Sensor Networks are composed by a large number of scattered sensor nodes that communicate with each other in order to obtain relevant information from the sensed area. The nodes can be scattered randomly through an area of interest, or be deployed into precise positions by human operators. Nevertheless, regardless how they have been deployed, the network must be always able to self-organize and forward the sensed data back to the end-user.

Wireless sensor networks can be found in many segments of human activity, such as farming, health, environment, or military forces. However, these networks are still struggling with a major problem, the scarce power source. Unlike wired sensor networks, in wireless environments, sensors cannot recharge their battery, which leads to the death of nodes, and consequently the death of the entire network. So, right now, it is important to increase, as much as possible, the network lifetime. To accomplish this goal, it is crucial to understand the tasks in which sensors spend more energy.

Presently, it is well known that sensor nodes consume more energy when they are transmitting messages. In other words, they spend more energy when they are transmitting messages than when they are sensing or processing data [33]. Furthermore, the consumption of power increases as the message's length, and the distance of communication increases. With regard to power consumption, any extra bit or extra meter counts since the death of few nodes may create blind spots or in the worst case scenario the death of the entire network.

In order to increase the lifetime of wireless sensor networks, the scientific community has made a huge effort, and because of that, many algorithms have been proposed, specially routing algorithms. However, none of them do what this work proposes.

This work presents you **LIASensor** (**L**ocality and **I**nterest **A**wareness for Wireless **Sensor** Networks), a project developed in the context of my MSc thesis. LIASensor is an improvement to environmental monitoring in wireless sensor networks and its main goal is to apply interest and locality awareness in wireless sensor networks.

In LIASensor, the locality awareness reduces the size of messages, whereas the interest awareness reduces the number of messages within the network. These two techniques are aimed to reduce the power consumed when a message is transmitted. As the number and size of messages decreases, so does the power needed to transmit the message. Thus, by reducing the power of transmission, the sensor can subsists a bit longer, which impacts positively the network lifetime.

This work is well suited to environments in constantly changing and networks on which the current data is more important than the past data. For instance, a researcher walking through a volcanic area; in such environment it is very important to know in which regions it is secure to enter, since small changes in the concentration of gases or the occurrence of seismic activity may indicate a possible situation of danger to the user.

LIASensor provides the user a detailed view about his surroundings, and an undetailed view about distant regions allowing the user to choose at every moment the path that matters most to the user.

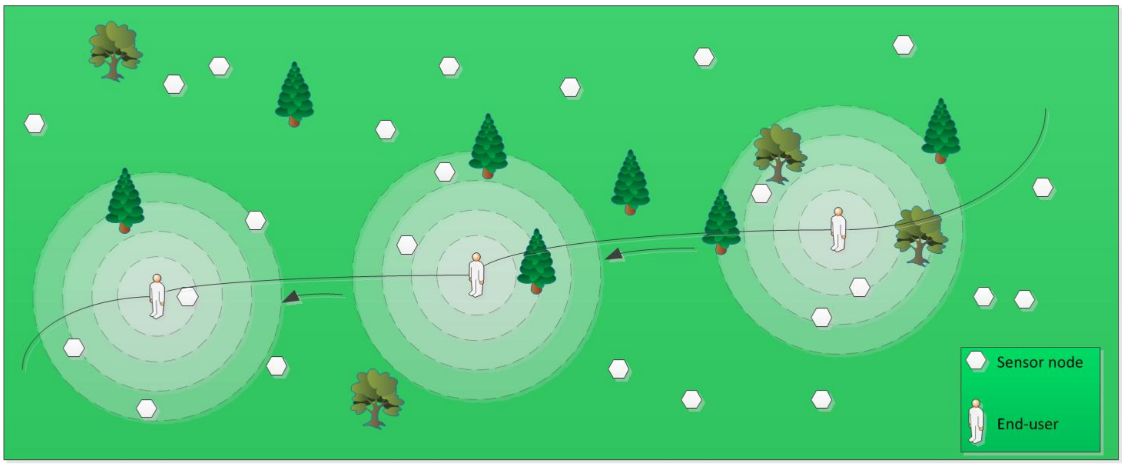


Figure 1.1: Example of an end-user walking through a WSN.

For instance, considering the figure 1.1, a user walks through a region covered with sensor nodes.

He does not need to follow predetermined paths or roads, i.e. he can walk freely throughout the region. As he progresses, his device disseminates queries, which are responded only by the sensors that received the packet. As the sensor's responses arrive to the user, the user's device represents a view of his surroundings. The zones closer to the user, the most opaque areas, represent the areas where he has more information, whereas the further regions, the most transparent areas, represent regions where the user does not have much information about it.

The quantity of information sent by each sensor depends of the distance that separates the sensor from the user, i.e. the quantity of information that reaches the user is determined as a function of the distance between the user and the sensor itself. This is accomplished through locality awareness. The type of information received by the user also varies regarding his interest, this is called interest awareness. In addition, the user can spread his radius of interest, increasing the query's range and the response's detail.

At last, it is important to note that the main concern of this thesis was the development of the prototype, as well as, the results of the simulation. So the device's layout was not prioritized. All the results, tests and simulations were executed and obtained through a simple console application.

1.1 Hypothesis and Methodology

In the context of my MSc thesis, I propose to test the hypothesis that through locality and interest awareness, it is possible to decrease the number of messages, as well as, the size of messages, and consequently increase the lifetime of the entire wireless sensor network. In order to test this hypothesis, I constructed a prototype system that implements both locality and interest awareness. The prototype was developed as a protocol and was executed in every single sensor node.

In order to test my prototype system and validate my hypothesis, I made some simulations with the purpose of assess and compare the network lifetime with and without my solution. The simulations were performed into a proper simulator to wireless networks.

1.2 Document Roadmap

The rest of this dissertation is organized as follows: Chapter 2 presents the theoretical foundations and the most important related work. Chapter 3 details the architecture and enforcement of

LIASensor. Chapter 4 presents the implementation details of the developed solution. Chapter 5 describes the experimental validation of the thesis proposal. Finally, Chapter 6 presents the main conclusions and discusses possible directions for future work.

Chapter 2

Concepts and Related Work

This chapter describes the fundamental concepts necessary to understand the research work that has been performed in the context of my MSc thesis. It also presents the most important related work, focusing on routing algorithms and techniques that are aimed to reduce messages in ad-hoc networks.

2.1 Fundamental Concepts

As mentioned in the first chapter of this dissertation, my MSc thesis relates to the area of interest and locality awareness in Wireless Sensor Networks. WSNs are an important area of investigation in distributed and ad-hoc networks. These networks are very important when we talk about automatic information retrieval, and information gathering in harsh environments. These networks are not straightforward; they can vary a lot, regarding their surrounding elements.

The following section introduces important concepts from Wireless Sensor Networks. Section 2.1.1 denotes and describes what is a sensor and how it works, while section 2.1.2 describes the meaning and structure of sets of sensors, i.e. Sensor Networks. Section 2.1.3 classifies Wireless Sensor Networks regarding they architecture, while section 2.1.5 describes the protocol stack within each sensor. Section 2.1.4 presents the factors that influence the performance of Wireless Sensor Networks. At last, section 2.1.6 introduces and organizes routing protocols for WSNs.

2.1.1 Sensor details

A sensor is a converter that measures a physical quantity and converts it into a signal which can be read by an observer or by an instrument. The output can be immediately read or be transmitted electronically over a network for reading or further processing.

A sensor node might vary in size from that of a shoebox down to the size of a grain of dust. The cost of sensor nodes is similarly variable, ranging from a few to hundreds of dollars, depending on the complexity of the individual sensor nodes. Size and cost constraints on sensor nodes result in corresponding constraints on resources such as energy, memory, computational speed and communications bandwidth

Sensors are used in everyday objects such as touchscreens displays (tactile sensor) and lamps which dim or brighten by touching the base. Beyond these, there are also innumerable applications for sensors of which most people are never aware, namely in cars, machines, and aerospace industry; they are also present in medicine, robotics, chemistry, environment monitoring, weather forecasting, acceleration and speed detection, distance calculation, and in many other modern applications.

2.1.2 Wireless Sensor Networks

According to [48], a WSN typically consists of a large number of low-cost, low-power, and multifunctional sensor nodes that are deployed in a region of interest. These sensor nodes are small in size, but are equipped with sensors, embedded microprocessors, and radio transceivers, and therefore have not only sensing capability, but also data processing and communicating capabilities. They communicate over a short distance via a wireless medium and collaborate to accomplish a common task, for example, environment monitoring, battlefield surveillance, and industrial process control. Wireless sensors have significant advantages over conventional wired sensors. They can not only reduce the cost and delay in deployment, but also be applied to any environment, especially those in which conventional wired sensor networks are impossible to be deployed, for example, inhospitable terrains, battlefields, outer space, or deep oceans.

2.1.3 Classification of WSN

Wireless Sensor Networks are applications specific. Usually, they are deployed for specific applications, so different networks have different characteristics. According to different criteria, WSNs are classified into different categories, as follows [48].

Static and Mobile Network: According to sensor nodes mobility, sensor networks can be considered static or mobile. In static WSNs all sensor nodes are static, they do not move. However, some sensor applications require mobile sensing nodes to accomplish their sensing task. For instance, monitor animals in their habitat. Compared with static networks the design of mobile sensor networks must consider the mobility effect, which is harder to implement, and therefore more complex.

Deterministic and Nondeterministic Network: WSNs can be either deterministic or nondeterministic. In deterministic networks the localization of each node is known; normally, in this kind of network, sensors are positioned manually. In Nondeterministic networks nodes are scattered randomly; these kinds of networks are placed in harsh or hostile environments. Compared to Deterministic networks, Nondeterministic networks are more scalable and flexible, though more complex too.

Static-Sink and Mobile-Sink Network: A data-sink is a device that sends queries or commands to the sensor nodes in the sensing region while the sensor nodes collaborate to accomplish the sensing task and send the sensed data to the sink and it can be either static or mobile. Static sinks are fixed in positions closer, or inside the network. A static sink makes the network easier to control; however, nodes near the sink tend to die first due the overload of messages that they have to forward. Mobile sinks change their position regularly, thereby balancing the traffic load among sensor nodes.

Single-Sink and Multi-Sink Network: Sensor networks can have a single sink or multiple sinks. In a single sink network, the sink can be located near or inside the network; in this topology every piece of sensed data is forwarded to the single sink. In multi-sink networks there are more than one sink, and therefore sensors can choose the closest sink to forward their data balancing effectively the traffic load among sensors.

Single-hop and Multi-hop Network: Sensor networks can be single-hop networks or multi-hop networks according to the number of hops between the sensors and the sink. In single hop networks each node communicates directly with the sink. In multi-hop networks, sensor nodes forward their data to the sink through nodes that are closer to the sink. The second approach is more energy-aware, and hence increase the lifetime of the whole network.

Homogeneous and Heterogeneous Network: Networks can be homogeneous or heterogeneous regarding the capabilities of their sensor nodes. In homogeneous networks all sensors are equal; every sensor has the same computational and transmission power as well as the same amount of energy. In heterogeneous networks there are sensors, usually a minority, more powerful than others. In this scenario the network can assign more processing and communication tasks to those sophisticated nodes in order to improve its energy efficiency, and thus increase the network lifetime.

2.1.4 Factors influencing WSNs

WSNs performance is influenced by many factors, such as *Production Costs*, *Power Consumption*, *Scalability*, *Adaptability*, *Reliability*, *Fault Tolerance*, *Security*, *Transmission Media*, *QoS*, *Support*, and *Self-Configurability*. The way some of these factors influence WSNs is addressed below [1, 2].

Production Costs: Sensor nodes are normally deployed in harsh and difficult access environments in large numbers. Therefore, sensor nodes are devices that cannot be recovered or reused. Due to these factors sensor nodes cost must be kept as low as possible, to make possible the creation of cheap and disposable networks.

Power Consumption: Node sensors being micro-electric devices have a limited power source, so wireless sensor networks lifetime depends of the battery lifetime. In a multi-hop ad-hoc sensor network, each node plays both the role of data originator and data router. The death of few nodes can cause significant changes in the topology of the network and therefore, decrease the lifetime of the WSN. Although, being very difficult to replace or recharge sensor batteries, recent advances in ambient energy harvesting technologies [31] have made possible for sensor nodes to rely on energy harvesting devices for power.

Scalability: Sensor nodes can be scattered in large areas and be in larger numbers, in order of tens, hundreds or thousands. Depending on the application, the number may reach an extreme of millions. Thus, network protocols designed for sensor networks should be scalable to different network sizes.

Adaptability: In a WSN a node can move, fail or join the network, which results in network topology changes. Hence, networks protocols should be easily adaptive to face these adversities and modifications.

Reliability: For many sensor network applications, it is required that data is reliably delivered over noise, error-prone, and time-varying wireless channels. To achieve the above-mentioned requirements, WSNs protocols must provide error control and correction mechanisms to ensure reliable data delivery.

Fault Tolerance: Some sensors may fail or be blocked due to the lack of power, may have physical damage or environmental interference. The failure of sensors should not affect the overall task of the sensor network. Thus, sensor nodes should have tools that guarantee fault tolerance.

Security: In some situations, sensor nodes are deployed in hostile environments, e.g. military applications, and thus are vulnerable to enemies. In such situations sensor nodes must apply effective security mechanisms [29] in order to prevent access from unauthorized users or malicious attacks.

Transmission Media: In a WSN, communication can be made through many technologies, such as radio, infrared, acoustic, optical media or bluetooth. Regardless the technology, the choice of transmission medium must be supported by robust coding and modulation schemes.

2.1.5 Protocol Stack

Like regular networks, WSNs also have a five layer protocol stack [1] divided into *physical layer*, *data link layer*, *network layer*, *transport layer*, and *application layer*, as figure 2.2 shows.

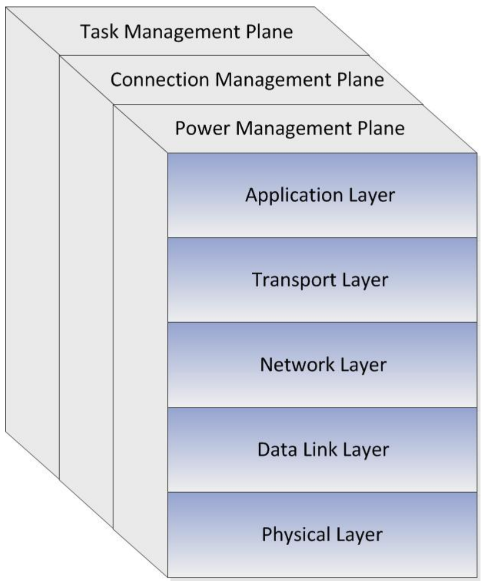


Figure 2.2: Protocol Stack for Sensor Networks.

- **Application Layer** - Application layer includes a variety of protocols that perform various sensor network needs, such as query dissemination, node localization, time synchronization, or network security;

- **Transport Layer** - In general, the transport layer is responsible for reliable end-to-end data delivery between sensor nodes and the sink;
- **Network Layer** - The network layer is responsible for routing the data sensed by source sensor nodes to the data sink;
- **Data Link Layer** - The data link layer is responsible for data stream multiplexing, data frame creation and detection, medium access, and error control, in order to provide reliable point-to-point and point-to-multipoint transmissions;
- **Physical Layer** - The physical layer is responsible for converting bit streams from the data link layer to signals that are suitable for transmission over the communication medium.

In addition to the five layers, the protocol stack can be also divided into three groups of management planes across each layer, including power, connection, and task management planes. **The power management plane** is responsible for managing the power level of a sensor node for sensing, processing, transmission and reception, which can be implemented by employing efficient power management mechanisms at different protocol layers. **The connection management plane** is responsible for the configuration and reconfiguration of sensor nodes to establish and maintain the connectivity of a network in the case of node deployment and topology change due to node addition, node failure, or node movement. **The task management plane** is responsible for task distribution among sensor nodes in a sensing region in order to improve energy efficiency and thus increase network lifetime.

2.1.6 Routing protocols in WSNs

Routing protocols are an important issue is wireless sensor networks, since energy preservation is one of the main concerns in these networks. The essential function of a WSN is to monitor a phenomenon in a physical environment and report sensed data to a central node called a sink. The transfer of information is made through paths discovered and defined by routing protocols. If the chosen paths are not the most energy aware, the nodes in the network will spend more energy than they actually need to forward the information, which will cause the premature death of the wireless network. So, this subsection focuses on routing protocols categorization according to their operation, structure and reaction.

According to [3], routing protocols in WSNs are divided into three categories: **flat-based routing**, **hierarchical-based routing** and **location-based routing** depending to network structure, as shown in figure 2.3.

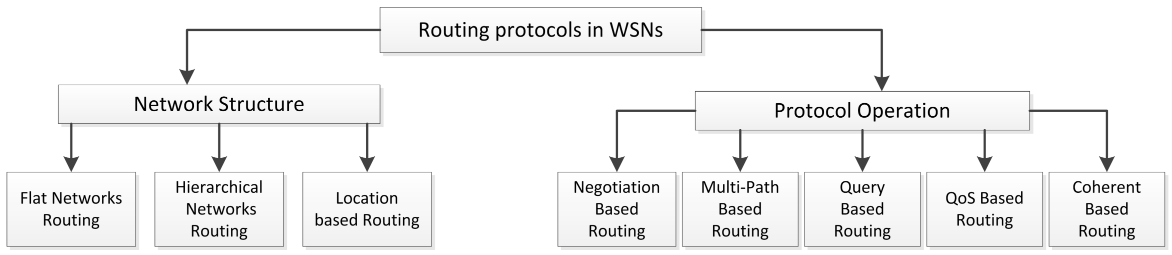


Figure 2.3: Routing Protocols in WSN: A taxonomy.

Flat-based routing: In flat architectures all nodes play the same role, and collaborate together to perform the sensing task. In this approach sensors forward their data to the sink directly or through multi-hop techniques. Due the large number of sensor nodes it is not feasible to assign global identifiers to every node. For this reason, data gathering is usually accomplished by using data-centric routing, where the base-station sends queries to certain regions and waits for data from the sensors located in the selected regions. Early works on data centric routing, e.g., SPIN [15, 20] and directed diffusion [16, 17] were shown to save energy through data negotiation and elimination of redundant data. These two protocols motivated the design of many other protocols which follow a similar concept, such as Rumor Routing [4], Cougar [43], and Home Agent Based Information Dissemination [35].

Hierarchical-based routing: Compared to flat-based algorithms, hierarchical architectures have huge advantages in scalability. As such, the concept of hierarchical routing is also utilized to perform energy-efficient routing in WSNs. In a hierarchical architecture, higher energy nodes can be used to process and send information to the sink while lower energy nodes can be used to perform the sensing task in the proximity of the phenomena area. A large number of algorithms and techniques using hierarchical routing have been proposed along the years. Section 2.2.1 explores in more detail these algorithms.

Location-based routing: In location-based routing, sensor nodes are addressed by means of their location. The distance between neighboring nodes can be estimated on the basis of incoming signal strength, or alternatively through Global Positioning System (GPS). There are several location-based protocols proposed for WSNs, Geographic Adaptive Fidelity (GAF) [42], Geographic, Energy-Aware Routing [46] and Span [7] are some examples.

Beyond the classifications addressed above, in accordance with [3] routing protocols can be also classified according with their operation, as shown in figure 2.3. Routing protocols can be classified into: *Negotiation-based routing*, *Multipath-based routing*, *Query-based routing*, *QoS-based routing* and *Coherent-based routing*. Contrary to network structure classification, these classifications are not exclusives; one protocol can be within one or more operational category.

In addition to the above-mentioned classifications, protocols can be classified into three exclusive categories, as shown in figure 2.4: *proactive*, *reactive* and *hybrid* protocols depending on how sensors respond to alterations in their sensing area.

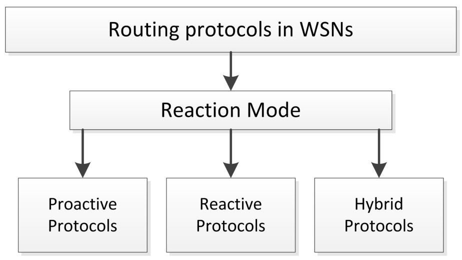


Figure 2.4: Routing Protocols in WSNs: Another taxonomy.

Proactive protocols: Nodes periodically switch on their sensing and transmitting components, sense the environment and transmit the data of interest. Hence, they provide regulars snapshots of the sensed world. They are well adapted for applications requiring periodic data monitoring.

Reactive protocols: Nodes react immediately to sudden and drastic changes in the value of a sensed attribute.

Hybrid protocols: Hybrid protocols use a combination of both protocols.

2.2 Related Work

This section presents the most relevant previous work in the context of my MSc thesis. All the work presented here tries to increase the lifetime of sensor nodes and consequently the lifetime of WSNs. Section 2.2.1 describes the most famous and relevant clustering algorithms, while section 2.2.2 presents a set of techniques able to reduce the amount of messages in ad-hoc networks.

2.2.1 Clustering algorithms

As mentioned in the last section, hierarchical protocols are the most scalable algorithms, and among this set of algorithms, cluster-based routing techniques represent the major part. The utilization of such clustering algorithms provides: 1) scalability enhancement; 2) communication efficiency and 3) possibility of data aggregation.

In cluster-based routing protocols, sensors are organized in regions known as clusters. Cluster members (just called sensor nodes) send their information to cluster-heads, which forward the information collected to the sink or base-station. Each node assigns itself to one cluster-head.

Depending of which algorithm is being executed; the number of clusters might be very different. Zero cluster-heads or 100% of cluster-heads is the same as direct communication. The number of cluster-heads can be assigned directly before the network being deployed or dynamically during the routing algorithm execution.

Cluster algorithms communications are made through the following two modes:

- **Intra-cluster communication** (forwarding to cluster-head): each sensor node sends the sensed data to the elected cluster-head;
- **Inter-cluster communication** (forwarding to base-station): each cluster-head sends data either to neighboring cluster-heads or directly to the sink whether it is near.

2.2.1.1 LEACH Low-Energy Adaptive Clustering Hierarchy

[13] presented LEACH, the first cluster-based algorithm proposed. The algorithm tries to distribute evenly the load through sensors in the network. To accomplish its purpose LEACH uses localized coordination to enable scalability and robustness for dynamic networks, and incorporate data fusion into the routing protocol to reduce the amount of information that must be transmitted to the base-station, further reducing energy dissipation and enhancing system lifetime. The protocol is based in two assumptions:

- The base-station is fixed and located far from the sensors.
- All nodes in the network are homogeneous and energy-constrained.

Thus, communication between the sensor nodes and the base-station is expensive, and there are no high-energy nodes through which communication can proceed.

According to the protocol, sensors elect themselves to be local cluster-heads at any given time with a certain probability. These cluster-heads nodes broadcast their status to other sensors in the network. After the status dissemination, each node determines to which cluster it wants to belong by choosing the cluster-head that requires the minimum communication energy. Once all the nodes are organized into clusters, each cluster-head creates and broadcasts a Time Division Multiple Access (TDMA) schedule for nodes within its cluster telling each node when it can transmit.

Once the cluster-head has all the data from its cluster-members, it fuses and transmits the data to the base-station. Since the base-station could be far away, this could be a high energy transmission. However since there are only a few cluster-heads, this only affects a small number of nodes.

The decision to become a cluster-head depends on the amount of energy left at node. In this way, nodes with more energy remaining will perform the energy-intensity functions of the network. Each node makes its decision about whether to be a cluster-head independently of the other nodes in the network.

- **LEACH details**

The operation of LEACH is broken up into rounds, where each round begins with a set-up phase, when the cluster is organized, followed by a steady-state phase, when data transfers to the base-station occur.

The decision for a sensor to become a cluster-head is made independently without any negotiation with the other sensors. Specifically, a sensor decides to become a cluster-head based on the desired percentage *P* of cluster-heads (determined a priori), the current round, and the set of sensors that have not become cluster-heads in the past 1/P rounds. If the number of cluster-heads is less than *T(n)*, a sensor n becomes a cluster-head for the current round, where *T(n)* is a threshold given by

$$T(n) = \begin{cases} \frac{P}{1-P(rmod\frac{1}{P})} & if n \epsilon G \\ 0 & othewise \end{cases} \tag{2.1}$$

In equation 2.1 *P* is the desired percentage of cluster-heads in the sensor population, *r* is the current round number, and *G* is the set of nodes that have not been cluster-heads in the last 1/P rounds. Using this threshold, each node will be a cluster-head at some point within 1/P rounds. During round zero (r = 0), each node has a probability *P* of becoming a cluster-head. The nodes that are cluster-heads in round 0 cannot be cluster-heads for the next 1/P rounds.

- **LEACH limitations**

LEACH was the first of its kind allowing a huge enhancement of efficiency and power saving when compared with previous routing protocols. However, it has some limitations and deficiencies:

- It assumes that all nodes can transmit with enough power to reach the base-station.
- It is not obvious how the number of the predetermined cluster-heads is going to be uniformly distributed through the network.

- Although LEACH solves the energy-balancing problem by rotation, it ignores the energy consumption issue of intra-clusters.
- Some very big clusters and very small clusters may exist in the network at the same time.

Regarding the problems of LEACH, that are described above, various modifications have been made to LEACH protocol: E-LEACH [41], TL-LEACH [26], LEACH-C [14], V-LEACH [44] and FZ-LEACH [18] are some examples.

2.2.1.2 HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad Hoc Sensor Networks

HEED [45] was based on LEACH and took into account the residual energy of nodes in electing cluster-heads. The algorithm was proposed with the following four major goals:

- Increasing network lifetime by distributing energy consumption.
- Terminating the clustering process within a constant number of iterations.
- Minimizing control overhead (to be linear in the number of nodes).
- Producing well-distributed cluster-heads and compact clusters.

HEED periodically selects cluster-head based on a hybrid of two clustering parameters: the primary parameter is the residual energy of each sensor node and the secondary parameter (unlike LEACH) is the intra-cluster communication cost as a function of neighbor proximity or cluster density.

The residual energy is used to probabilistically select an initial set of cluster-head while the intra-cluster communication is used to choose on which cluster-head the node will be assigned when it falls within the ”range” of more than one cluster-head.

The HEED clustering improves network lifetime when compared with LEACH because LEACH randomly selects cluster-head (and hence cluster sizes), which may result in faster death of some nodes. The final cluster-head selected in HEED is well distributed across the network and the communication cost is minimized.

2.2.1.3 Energy Adaptive Cluster-Head Selection for WSN

Another improvement for LEACH was presented by Liang [22]. The algorithm modifies LEACH cluster-head selection by adding additional parameters into LEACH including the remaining energy level of candidates and their energy costs for transmission. The main idea of the algorithm is to avoid choosing nodes with lower residual energy and higher energy dissipation as cluster-heads. It can be achieved by making some beneficial adjustments to the threshold T(n) proposed in LEACH.

2.2.1.4 Threshold Sensitive Energy Efficient Sensor Network Protocol (TEEN)

TEEN [27] was the first protocol developed for reactive networks. Each cluster-head broadcasts to their members two values: the *hard threshold* and the *soft threshold*. The hard threshold is a value for the sensed attribute, beyond which a sensor should turn its transmitter on to report its sensed data to the cluster-head. The soft threshold indicates the acceptable difference between two followed sensed values, i.e. if a node senses two straight values with a difference bigger than the soft threshold the node turn on its transmitter and transmit the sensed value to the cluster-head. As can be seen, the hard threshold reduces the number of transmissions by allowing the node to transmit only when the sensed value is in the range of interest; the soft threshold further reduces more the number of transmissions by eliminating transmissions with little or no change in the sensed value.

Hence, TEEN allows that: 1) time critical data reaches the user almost instantaneously; 2) energy saving, even though the nodes sense the environment continuously, energy consumption is much larger when the nodes are transmitting; 3) adjustments in the soft threshold gives more accurate data.

The main drawbacks of this protocol are that, if the thresholds do not reach the nodes, they will never communicate their sensed value. Besides this problem, the values of thresholds must be set carefully to keep the sensors responsive.

To overcome the above-mentioned issues, a new protocol, called adaptive periodic TEEN (**APTEEN**), was proposed. APTEEN [28] is a hybrid protocol that combines the best features of both LEACH and TEEN. The nodes of the network not only react to time-critical situations, as TEEN, nut also give an overall snapshot of the world at periodic intervals of time, as LEACH does.

2.2.1.5 PEGASIS: Power-Efficient Gathering in Sensor Information Systems

PEGASIS (Power Efficient Gathering in Sensor Information Systems) [23] presented an enhanced LEACH protocol with a chain topology. Unlike LEACH, PEGASIS avoids cluster formation and uses only one node in a chain to transmit to the base-station instead of using multiple nodes. The chain topology on which PEGASIS works can be performed either by the sensors themselves, using a greedy algorithm, or by the sink, which has to broadcast the chain to all sensors in the network. For constructing the chain, it is assumed that all the nodes have global knowledge of the network. When a node dies, the chain is reconstructed in the same manner. The chain has two end sensors and in each data fusion round only one is designated leader. The leader node is responsible for transmitting the fused data to the sink and is selected randomly in each round.

For gathering data, each node receives data from its neighbor, fuses the data received with its own data and forward the data to other neighbor on the chain.

Nevertheless, PEGASIS introduces excessive delay for distant nodes on the chain. Besides, one single leader, transmitting data to the sink, becomes a bottleneck. To solve this problem an extension to PEGASIS was created. Hierarchical-PEGASIS [24] aims at solving the delay problem. In order to reduce the delay in PEGASIS, Hierarchical-PEGASIS allows simultaneous transmissions of data messages to the sink.

2.2.1.6 Clustering Algorithm Based on Communication Facility (CABCF)

In CABCF [25] both inter-cluster and intra-cluster communications are made in a multi-hop way. The algorithm elects Cluster-heads considering: 1) the communication facility (CF), 2) the distance between nodes and the base-station, 3) the degree (measured by the number of nodes that transmit to the sink through that node), and 4) the residual energy of each node and the cluster size.

Communication facility is the comprehensive measurement of two factors:

- The convenience of the node communicates directly with the sink;
- The convenience, of through it, one node let another communicate with the sink.

The idea of clustering algorithm based on communication facility is that each node calculates its own communication facility; then, each node combines itself with its neighbor nodes; when all the nodes are combined within a zone, CABCF selects the node with the highest CF as cluster-head; cluster-heads, then change information in order to generate inter-cluster communications. Intra-cluster communication is made in a multi-hop way according to the process of cluster formation;

inter-cluster communication is made from the lower-CF node to the higher-CF node.

When the energy of one of the cluster-heads reaches a level below a predetermined value, it advises the sink, which triggers a new round of cluster formation.

2.2.1.7 An Energy-Efficient Protocol with Static Clustering for WSN (EEPSC)

EEPSC [47] is an algorithm that partitions the network into distance-based static clusters, eliminates the overhead of dynamic clustering and utilizes temporary cluster-heads to distribute, evenly, the energy load among sensor nodes.

EEPSC forms clusters only once, during the network lifetime; for this aim, the base-station broadcasts *k-1* different messages with different transmission powers, where *k* is the desired number of clusters. All sensor nodes that hear the message set their ID to *k* and inform the base-station that they are members of the cluster *k*, by transmitting a join-request message back to the base-station. Afterward, the base-station selects randomly one temporary cluster-head for each cluster.

After the clusters are formed, the network starts its normal operation and the selection of cluster-head starts. At the beginning of each round, every node sends its energy level to the temporary cluster-head. Afterward, the temporary cluster-head: 1) choose the sensor node with more energy level as cluster-head for the current round, which will collect the data, perform local data aggregation and communicate with the base-station; and 2) choose the node with lowest energy level as temporary cluster-head for the next round.

EEEPSC [5] (Enhanced Energy-Efficient Protocol with Static Clustering) is a modification of (EEPSC). Similar to EEPSC, the algorithm partitions the network into distance-based static clusters. However, unlike EEPSC, cluster-head selection is performed by taking into account both the spatial distribution of sensors nodes in the network and their residual energy with the objective of reducing the intra-cluster communication overhead among the nodes and thus making the scheme more energy-efficient.

2.2.1.8 Local Negotiation Clustering Algorithm (LNCA)

LNCA [39, 40] introduced the concept of **Isoclusters**. In an Isocluster, the node readings are very close in value, and they can be easily compressed by the cluster-head. Consequently, the overall volume of data sent to the sink is greatly reduced.

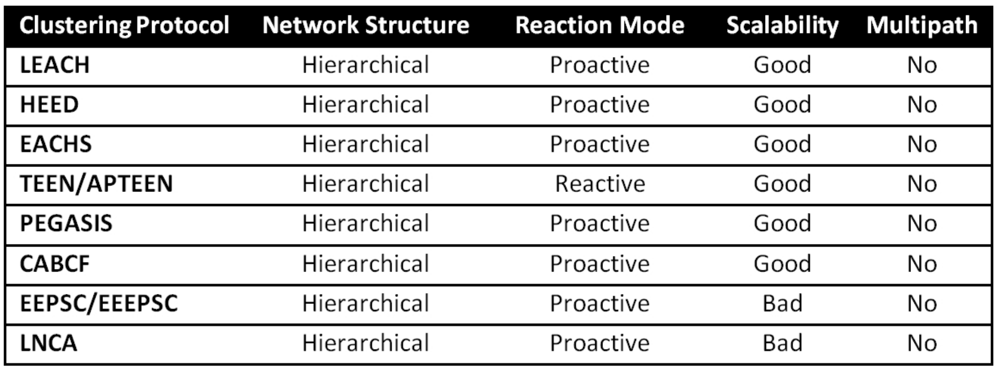
This approach employs the similarity of nodes readings as the main criterion in cluster formation. The technique also tries to avoid creation of excessive number of clusters by generating n-hop clusters.

The algorithm forms clusters in a distributed manner through self-organization of the sensor nodes; the cluster-head are distributed equally throughout the WSN, so any two cluster-head are neighbors.

However, this approach has some limitations; LNCA uses IDs to identify each sensor, which in large WNS, with several hundred or thousands of sensors causes massive overhead. Another limitation shows up when the communication with the sink is made at large distances, thus it is expectable that those cluster-heads might be the firsts discharging and consequently ”killing” the whole network.

2.2.1.9 Conclusions

This section presented the most relevant and important clustering algorithms. However, there are still existing many others clustering algorithms. For instance, [38] proposes the use of gateways nodes in each cluster in order to reduce the transmission radio of the cluster-head achieving that way better power saving; [34] improves LEACH by modifying the LEACH set-up phase; region-based Energy-aware Clustering (REC) [12] proposes a scheme to efficiently create cluster, select cluster-heads and perform inter and intra cluster communication; [10] chooses the cluster-head in the same gradient grade; EBLEC [6] is a self-organizing, static clustering scheme in which clusters are formed only once during the network's lifetime. One critical step in node clustering is to select a set of cluster-head and group the remaining sensor nodes into clusters with these cluster-head. Table 2.1 summarizes, and makes a comparison among the presented protocols.



Clustering Protocol	Network Structure	Reaction Mode	Scalability	Multipath
LEACH	Hierarchical	Proactive	Good	No
HEED	Hierarchical	Proactive	Good	No
EACHS	Hierarchical	Proactive	Good	No
TEEN/APTEEN	Hierarchical	Reactive	Good	No
PEGASIS	Hierarchical	Proactive	Good	No
CABCF	Hierarchical	Proactive	Good	No
EEPSC/EEEPSC	Hierarchical	Proactive	Bad	No
LNCA	Hierarchical	Proactive	Bad	No

Table 2.1: Comparison between the Clustering-based protocols presented.

Node clustering is very important in WSNs because it provides a topology control approach to reduce transmission overheads and exploit data aggregation among a large number of sensor nodes. However, it does not considerers the end-user's interest and locality. Since these algorithms are network layer protocols, their main task is guarantee the effectively delivery of messages. So, it still necessary protocols above clustering algorithms capable of figure out which messages and information is important for the end-user.

2.2.2 Message Reduce Techniques in Ad-Hoc networks

Besides enhancing the network lifetime through more efficient routing protocols utilization, it is absolutely necessary to reduce message's length and number. To achieve this feature some techniques have been developed. Data fusion, which is currently used and Vector Field Consistency, a recently model, but with many promising applications are two examples.

2.2.2.1 Interest Management

Interest Management is a set of techniques able to filter and dispose relevant information to entities that are interested in it.

Interest Management can be abstracted using a *publish-subscribe* model. In such models, *Publishers* are objects that produce events, *Subscribers* are objects that consume events, and an object can be both a publisher and subscriber.

Without *Interest Management* a receiving entity would receive messages from every producing entity; so, the receiving entity would be responsible for sorting through and discarding useless messages. The concept of *Interest Management* was developed to address this problem by reducing the arriving messages to a smaller and relevant set. Under *Interest Management*, an entity expresses its data interests in terms of location and other application-specific attributes. According to Morgan [30], other agents in the simulation infrastructure, *interest managers* (*IMs*), accept entities' *interest expressions* (*IEs*) and use them to filter messages to sets (or reduce supersets) which meets the entities' needs.

An *Interest Expression* (*IEs*) is a specification of the data one entity needs to receive from other entities in order to interact with them correctly. *IEs* may refer to several attributes of the simulation entities or to the radius of interest, i.e. an entity may express interest about all entity within a 50 meters radius around itself.

Morgan [30] develop a three-category taxonomy for classifying *Interest Management* schemes: *Communication Model*, *Filtering Focus*, and *IM Domain of responsibility* (*DOR*).

- **Communication Model** - The three types of communication model are *unicast*, *broadcast*, and *multicast*. Broadcast means transmitting the same data to all possible destinations. Multicast is the term used to describes communication where a piece of information is sent from one or more points to a set of points. Unicast is the term used to describe communication where a piece of information is sent for one point to another point.

- **Filtering Focus** - Filtering Focus differentiates systems that filter data based on *intrinsic* characteristics of objects, such as the values of their attributes, from those that filter data based in *extrinsic* characteristics of objects, such as their location in a cell.
- **IM Domain Of Responsibility (DOR)** - An IM may have either *static* or *dynamic* DORs. An IM's DOR is the area in multi-dimensional parameter space in which the IM is responsible for managing data transmission. A dynamic DOR may change size and shape as well as location, whereas in static DORs IMs are assigned to predefined grid cells, entities may move in and out of the IMs' DORs, but the DORs themselves don't change.

2.2.2.2 Vector Field Consistency

Vector Field Consistency (VFC) [37] is a client-server architecture based on interest and locality awareness techniques. Furthermore, VFC is an optimistic consistency model allowing bounded divergence of object replicas.

In VFC, the server is the responsible for keeping the actual state of the world, and regularly updates clients. Each client registers within the server the objects, also called pivots, that will be shared, as well as, their consistency parameters, which consist in a 3-dimmensional vector.

By analogy with the electric $\vec{E}$ and the gravitational $\vec{G}$ fields, a pivot generates a consistency field determining the consistency of each object as a function of the distance between the object and the pivot. Thus, pivots generate consistency zones, iso-surfaces, ring shaped, concentric areas around them, such that the objects positioned within the same consistency zone are enforced the same consistency degree, For example in Figure 2.5 pivot P is in the center of four consistency zones labeled z_i, where $0 \leq i \leq 4$. Objects o_2 and o_3 are enforced the same consistency degree since they are in z_3.

The pivot's surroundings are updated according with its consistency degree, an object closer has a higher consistency degree than an object further, and therefore its update message is more detailed and complete. Higher consistency degrees mean greater details and higher update rate.

Consistency degrees are defined through 3-dimmensional vector, which specifies: 1) the maximum time a replica can be without being refreshed with its latest value, 2) the maximum number of lost updates replicas, i.e., updates that were not applied to a replica, and 3) the maximum

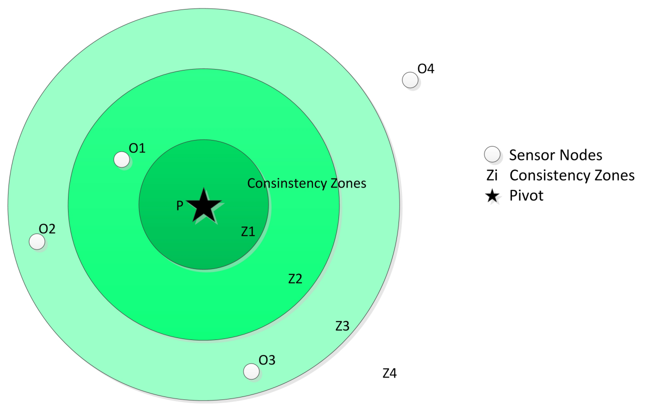


Figure 2.5: Conceptual consistency zones.

relative difference between replica contents or against a constant.

- **VFC over Ad-hoc Networks**

In ad-hoc networks there is a need for data sharing between the network nodes. Enforcing data consistency requires additional communication for update propagation and synchronization operations. However, communication-intensive operations are critical and have a twofold negative impact, Firstly, the high latency, the reduced network bandwidth and the small processing capability of devices brings overhead and dramatically hinders game playability. Secondly, extensive access to the network causes devices batteries to consume rapidly.

M. Santos [36] and N. Santos [37] combine the VFC consistency model with middleware to multiplayer games in ad-hoc networks. In both works upon the establishment of the ad-hoc network, one of the nodes becomes the server. Naturally, the server device also acts as client allowing all nodes to participate in the game. The server coordinates the propagation of updates to clients according to the VFC consistency parameters specified by each client.

The advantages of VFC are manifold. It is flexible and easily perceived by game programmers: the consistency model based on pivots is intuitive and the parameterization settings allow the game programmer to specify the consistency requirements for a wide range of game scenarios. Also, by intelligently selecting the critical updates to send and postponing the less critical ones, VFC is efficient in the utilization of resources, it reduces network bandwidth usage and masquerades latency.

2.2.2.3 Data fusion

Data fusion, also called, data aggregation is the process of integration of multiple data and knowledge representing the same real-world object into a consistent, accurate, and useful representation. The expectation is that fused data is more informative and synthetic than original inputs.

Applying data fusion to ad-hoc networks, specifically WSNs is very beneficial to the whole network due the decreasing of redundant information. Sensor nodes are usually deployed in large or even huge number in systems or areas of interest. Because of dense pattern of sensor deployment, neighboring sensor nodes may sense similar data on specific phenomenon. Since sensor nodes are run by battery power, it is critical to perform every operation in an energy-efficient manner. For this purpose, it is desirable for a sensor node to remove the redundancy in the data received from its neighboring nodes before transmitting the final data to the sink. Data aggregation is an effective technique for removing data redundancy and improving energy efficiency in WSNs. The basic idea is to combine the data received from different sources so that the redundancy in the data is minimized and the energy consumption for transmitting the data is reduced. The data-centric nature of WSNs makes data aggregation a crucial task [19].

Different authors consider different fusion methods. Li [21] considers five representative fusion methods. However, Zheng [48] considers there are only three major data aggregation techniques.

2.2.2.4 Conclusions

One of the most important constraints on sensor nodes is the low power consumption requirement. The power sources for sensor nodes are limited and generally irreplaceable. Therefore, while the objectives of traditional ad-hoc networks focus on high QoS provisioning, WSNs must focus primarily on power conservation. It is important to exploit the trade-off between longer network lifetime and lower throughput or higher transmission delay.

This section presented Data fusion and Vector Field Consistency. VFC is a pioneer technique based in interest-awareness and locality-awareness, which is capable of reducing messages traffic. On the other hand, data fusion is a technique capable of reduce both number of messages and messages' length; nevertheless its use is only possible under certain circumstances.

2.2.3 Distance calculation

Accurate and low-cost sensor localization is a critical requirement for the development of wireless sensor networks in a wide variety of applications. There are some techniques suited to locate nodes; however, all of them have pros and cons. [32] described measurement-based statistical models useful to describe many of these methods. Among the location techniques, these are the most used:

- **Time of Arrival (TOA)** - Calculates the distance through time and signal propagation velocity. It is the travel time of a radio signal from a single transmitter to a remote single receiver. Since TOA relies on the difference between the time of arrival and time of departure, all receivers and transmitters must be synchronized so there is no error in the difference due to clock offsets. This may prove to be a problem, especially considering the high speed at which the signals travel. Also, as with any time sensitive systems, there is also the possibility of significant hardware delays that must be accounted for to calculate the correct distances
- **Angle of Arrival (AOA)** - Calculates the distance by getting the signal direction send by the adjacent node through the combination of array antenna and multiple receivers;
- **Received Signal Strength Indication (RSSI)** - Calculates the distance through the strength of the received signal.

A global positioning system (GPS) receiver on each device is a good solution to the future, but at this moment it still monetary and energy prohibitive for many applications.

2.3 Conclusions

This chapter presented some fundamental concepts and the state of art of wireless sensor networks. Although WSNs exists for several years, locality and interest awareness have not yet been applied to it. In fact, as mentioned above, much of the work done to increase the network lifetime was done through the development of new routing algorithms, mainly hierarchical algorithms. In addition, current locality and interest awareness techniques were not developed to wireless sensor networks need.

So, none of the techniques and algorithms presented in this chapter completely suits this thesis goal. Cluster-based algorithms are very efficient and they can increase the network lifetime; however, they main goal is reduce the routing cost. Data fusion only can be performed upon the environment sensing. VFC is well suited for replicated systems; however, its adaptation to WSNs is not straightforward.

Currently there are any algorithms or techniques capable of applying Interest and locality awareness in Wireless Sensor Networks. The closest approach was proposed by [11]. They have proposed a new message dissemination technique regarding mobile sinks.

In their solution, sensor nodes periodically update the sink; when a sensor has relevant information, it starts by forwarding to the sink (through flood) a route request; upon receiving three or more routing requests, the sink starts to move toward the sensor in order to decrease the distance between them. A number of routing requests less than three means the sink is close enough to receive the sensed data. At the same time, the sink updates the sensors with its new position. This solution has obvious problems and does not suit our goals; for instance, a large area implies a large dislocations made by the sink.

The next sections present a solution that allows the incorporation of locality and interest awareness in wireless sensor networks. The algorithm was called LIASensor (**L**ocality and **I**nterest **A**wareness for Wireless **Sensor** Networks).

Chapter 3

Architecture

This chapter presents the LIASensor architecture. It describes how the addition of interest and locality awareness in messages exchange allows LIASensor to reduce the number and the size of messages in WSNs. It also demonstrates the use of a client-server model. Besides, it also presents a finite state machine (FSM), which is the way that LIASensor works. And, at last, it presents a protocol stack, which permits the LIASensor to run over any kind of routing or mac protocol, without significant changes in its architecture or source code.

More specifically, this chapter is divided as follows. Section 3.1 presents sensor's internal structure. Section 3.2 describes LIASensor's client-server architecture. Sections 3.3 and 3.4 present locality and interest awareness enforcement. Then, section 3.5 describes LIASensor protocol stack. Section 3.6 presents LIASensor as a Finite State Machine, and at last, section 3.7 describes the way LIASensor calculates the distance between the end-user and the sensor nodes.

3.1 Sensor Structure

Each sensor node has a multi-layer protocol stack (section 2.1.5) responsible for forwarding the data to the end-user. Each layer executes different services and different protocols. Regarding the nature of the task, the protocols executed within each layer can vary a lot. However, their final purpose is always the same. LIASensor protocol stack is compound by:

- **Application Layer** - This is the layer where protocols are more diversified. Application protocols separate functionality above the transport layer;

- **Transport Layer** - This layer contains routing protocols, which is responsible for discovering and using the best routes between the nodes. Protocols in this layer are very important, since they play a major role in energy conservation;
- **Network Layer** - This layer is responsible for manage the Media Access Control (MAC) protocols, which provides addressing and channel access control mechanisms that make it possible for several network nodes to communicate within a multiple access network that incorporates a shared medium;
- **Data Link/Physical layer** - This layer consists of networking hardware transmissions technologies, such as bluetooth, WI-FI or infrared.

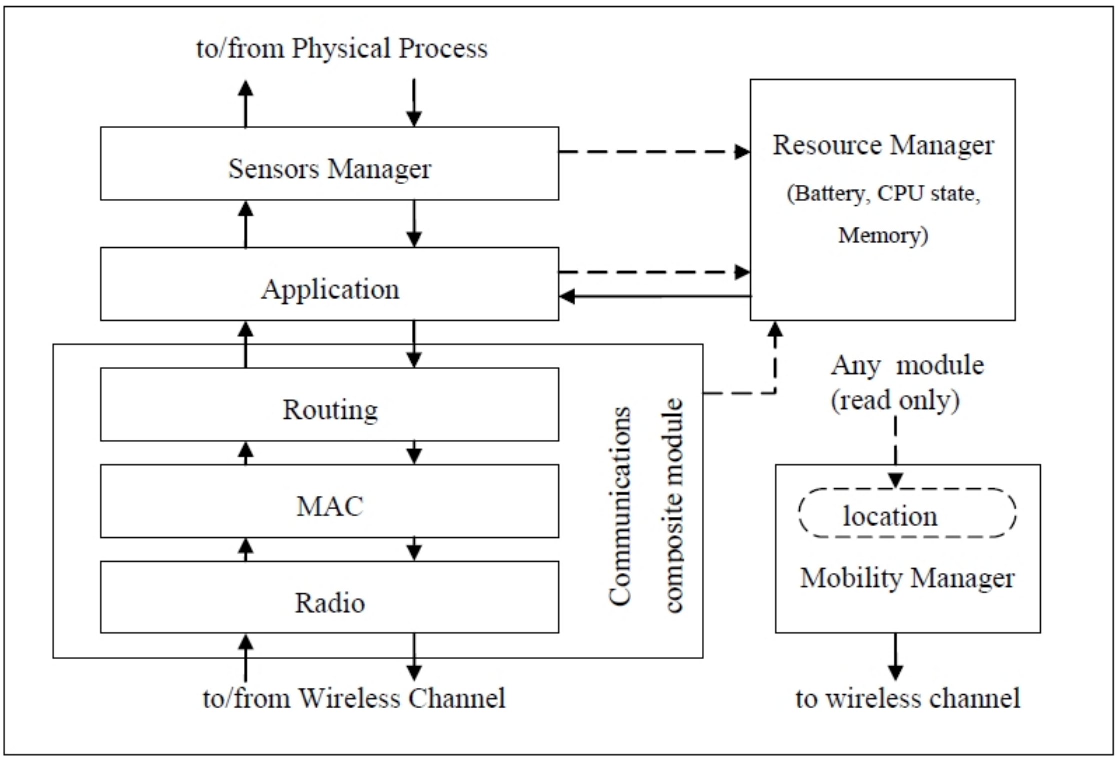


Figure 3.6: Sensor Structure.

Figure 3.6 represents sensors' internal structure. Each layer communicates directly with its adjacent layers. The solid arrows signify message passing and the dashed arrows signify simple function calling. The Radio, Mac, Routing and Application boxes represent respectively the Physical, Network, Transport and Application Layer in the protocol stack. The Physical Process Interface is the component by which sensors sense the environment.

LIASensor runs in the Application Module, over the communications composite module. When a node receives a request, a packet is created and sent to the wireless channel which then decides which nodes should receive the packet. Every LIASensor packet is created and read within the Application layer. For instance, when a query arrives the packet is forwarded up to the application layer to be processed; if a response is needed, a new packet is created and forwarded down to the Communications composite model. The communications composite module encapsulates the three bottom layers. This module is responsible for executing the radio, mac and routing protocols.

The Resource Manager is part of every sensor, and is responsible for monitoring sensors' resources, namely Battery, CPU state and Memory. The last module is the Mobility Manager, which permits sensors to move. This module is present in every sensor node; however it was not used in the context of this work.

3.2 Client-Server Architecture

Users and sensor nodes communicate with each other as a classic client-server model. In such models, clients are the machines that request services and functions, and servers are the machines that perform the task (the machines that offer the service). The client is responsible to initiate contact with servers in order to use their resources.

In LIASensor sensors play the role of servers, whereas end-users play the role of clients, as figure 3.7 represents. Client-server architecture has four types of relationships. Nevertheless, LIASensor only uses two of them: **many** to **one**, and **many** to **many** relations.

The many to one relationship (figure 3.8) is characterized by the existence of a plural number of servers and a single client. In LIASensor this means that at each moment in the WSN it will be deployed more than one sensor node and only a single end-user. On the other hand, the many to many relationship (figure 3.9) implies the existence of many servers, as well as, many clients too. This relationship in LIASensor is accomplished through the deployment of many sensors as well as the existence of many end-users. Each user queries the network independently and it is up to each sensor to correctly answer the request. Though, the many to many relationship contemplate the existence of many end-users, it is crucial that in each point of time the number of sensor nodes suppresses the number of user-ends.

To sum up, the end-user requests sensors' services, through the dissemination of queries in the WSN. A sensor receives the request, produces a response and disseminates the packet again in the WSN. The client-server model is set automatically without the need to do configurations or adaptations.

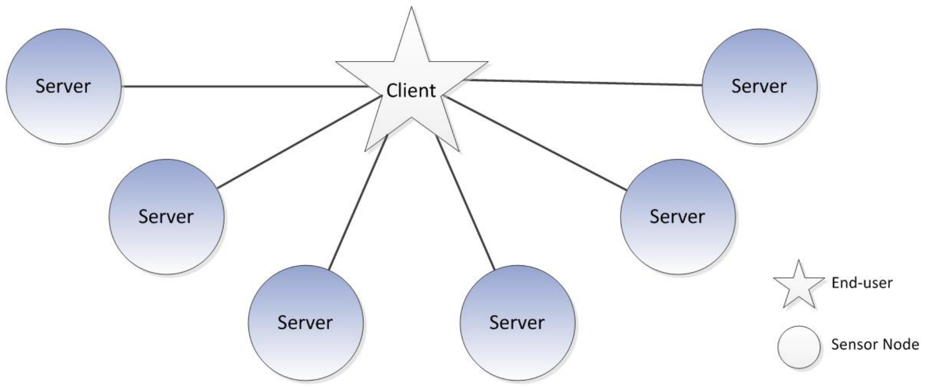


Figure 3.7: Node distribution in LIASensor.

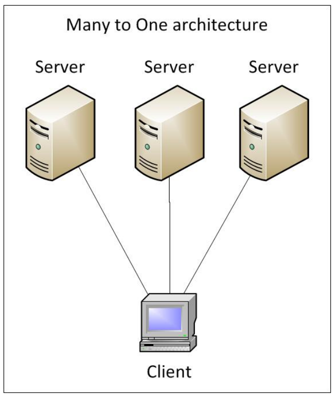


Figure 3.8: Many to One relationship.

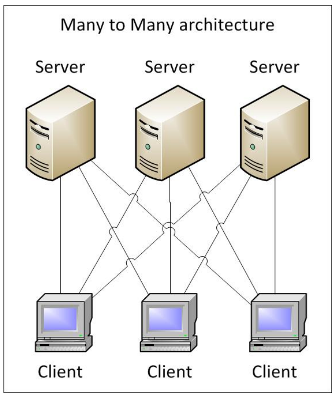


Figure 3.9: Many to Many relationship.

3.3 Locality-awareness

Locality awareness is a set of methods and techniques skilled to do information filtering. Its purpose is to guarantee that users only receive data within their region of interest. For instance, if a user manifests interest in a region with a radius of interest of 100 meters, locality-awareness techniques guarantee that he only receives data originated from sensors within those 100 meters. All the information originated from outside of those 100 meters will be silently dropped.

To accomplish Locality-awareness LIASensor generates radius of interest around each user through the signal strength of the user's device. Once a signal is generated, electromagnetic waves travel through space either directly, or have their path altered by reflection, refraction or diffraction. The intensity of these waves diminishes due to geometric dispersion, i.e. a more powerful signal reaches larger distances and a weaker signal reaches smaller distances, creating respectively larger and smaller radius of interest. In addition, since the signal is variable (the end-user can increase and decrease the signal strength) the user can create different radius of interest as he needs.

LIASensor uses this concept and by analogy with the electric $\vec{E}$ and the gravitational $\vec{G}$ fields generates consistency fields determining the consistency of each sensor node as a function of the distance between the sensor and the user, also called pivot. Thus, pivots generate consistency zones, iso-surfaces, ring shaped, concentric areas around them, such that the objects positioned within the same consistency zone are enforced with the same consistency degree, For example in Figure 2.5 a user P is in the center of four consistency zones labeled z_i, where $0 \leq i \leq 4$. Objects o_2 and o_3 are enforced with the same consistency degree since they are in z_3. Object o_1 has a higher consistency degree than objects o_2 and o_3 since it is closer to the pivot.

With locality-awareness techniques sensors respond according to their consistency degree. Nodes closer to the end-user naturally have higher consistency degrees, while nodes further have lower consistency degrees.

The concept of consistency degree is used to reduce messages' size. According to the calculated consistency degree, sensors respond more or less accurately, i.e. sensors with higher consistency degrees are more accurate than those with lower consistency degrees. This feature allows the user, who is patrolling, to know more details about his surrounding regions. Messages from distant regions are received too, but since those messages came from farther regions, they are less detailed.

In order to know in which consistency degree they are, sensors calculate the distance between them and the end-user. The distance can be calculated through many ways (section 2.2.3); however due to energy restrictions LIASensor uses the less energy-expensive.

3.4 Interest-awareness

As locality awareness, interest awareness is also a set of methods and techniques capable of information filtering. The use of these techniques guarantees that the end-user only receives information on which he demonstrates interest.

As we know, sensor nodes may contain many and different functions, and each one of them might have a different ability (section 2.1). LIASensor does not impose restrictions to their abilities. In fact, LIASensor only cares about the communication format; as long as the communication respects the protocol structure, every kind of sensor is permitted.

In LIASensor, interest-awareness is achieved through the specification of which type of information the user wants. Upon the construction of the query, the user specifies which information he wants to fetch from the environment. This is a simple method, but capable of reducing the amount of useless information in the WSN.

3.5 Protocol Stack

Considering the protocol stack (subsection 2.1.5), LIASensor can be executed as an application layer protocol or as an add-on to other application layer protocols.

When LIASensor works as a full application layer protocol, it can be executed on top of every protocol existing in bottom layers; thus, LIASensor is able to run over any kind of routing algorithm, such as flat, clustering or location based routing protocols without compromising its performance. The adaptation to these protocols is automatic without the need to do configurations or adaptations.

As an add-on algorithm to other application layer protocols (as figure 3.10 shows), LIASensor needs to be adapted to the application protocol since these protocols are not suited to interact with upper layers. Moreover, different adaptations are necessary to different protocols since the differences between application layer protocols are not few.

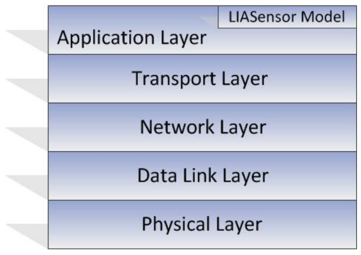


Figure 3.10: LIASensor has an add-on to application layer protocols.

3.6 Finite State Machine (FSM)

A finite state machine (FSM) is conceived as an abstract machine that can be in one of a finite number of states. The machine is only at one state at a time, and the state it is in any given time is called the current state. The machine can change from one state to another when initiated by a triggering event or condition; this is called a transition. The machine starts in an initial state and moves through a series of states as is fed by events and conditions. A particular FSM can be defined by:

- An initial state;
- A set of possible input events;
- A set of new states that may result from the input;

- A set of possible actions or output events that result from a new state.

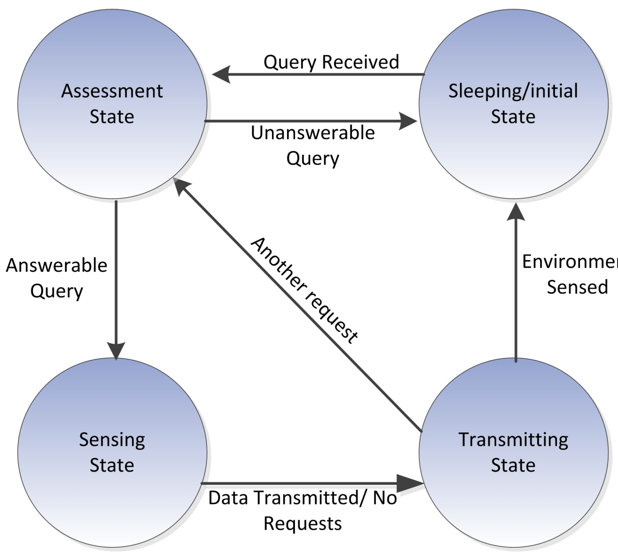


Figure 3.11: State Machine Representing the sensor nodes.

Considered as an abstract model of computation, the finite state machine is weak. However, it consumes less computational power energy than others models.

LIASensor works as a finite state machine. Each sensor is at one state at any given time. The list of states contains four different states (sleeping, assessment, sensing and transmitting states), and the list of inputs contains six elements (query received, unanswerable query, answerable query, no requests, another request and environment sensed).

Considering the FSM (figure 3.11), every sensor begins at the sleeping state (the initial state). Upon the occurrence of the event (the query arrival) the sensor switches from the sleeping state to the assessment state. At the assessment state, the sensor checks whether the query is addressed to it or not. The assessment is based in both interest and locality awareness techniques. If the assessment returns a positive value, i.e. the sensor can answer the query, the sensor transits to the sensing state. Otherwise, the sensor gets back to the sleeping state.

At the sensing state the sensor senses and stores environmental data. When the sensing task is completed, the sensor switches from its sensing state to the transmitting state. At the transmitting state the sensor constructs a packet compound by the actual sensed data and its previous records, and then sends the information to the end-user. When it finishes the transmissions task, the sensor has two different paths regarding the triggering event. If a new query event is triggered the sensor get back to the sensing state, otherwise the sensor finishes the cycle and returns to the initial state again (the sleeping state). Table 3.2 summarizes the FSM operation.

Current State	Input	Next state
Sleeping State	Query received	Assessment State
Assessment State	Unanswerable query	Sleeping State
	Answerable query	Sensing State
Sensing State	No requests	Transmitting State
Transmitting State	Environment sensed	Sleeping State
	Another request	Assessment State

Table 3.2: LIASensor state transition table.

3.7 Distance Calculation

The way of calculating the distance between the sink and the sensor nodes is very important, since this is the way that consistency degrees are generated. As chapter 2.2.3 has shown, there are three main ways of calculating this distance.

Sensors can also be equipped with GPS components. Nevertheless, the utilization of such component greatly increases the consumption of energy. LIASensor uses RSSI to calculate the distance between sensors and users. Despite not being an exact model, it is the most reliable among the other energy-aware models.

3.8 Conclusions

LIASensor architecture was built In order to respond to the specificities of this work. Any changes in the requirement assessment could lead to changes in the prototype architecture. Thus, the main goal of this architecture is to maximize the network lifetime in environments such as those described in the earlier chapters.

Interest and locality awareness techniques were used in a manner capable to decrease the number and size of messages. However, the most specific, and possible the most well-suited component of LIASensor is the finite state machine. The FSM represents the way that LIASensor works, and cannot be replicated or used directly in other systems, since it was developed to fit LIASensor requirements.

On the other hand, the less specific component of LIASensor's architecture is the sensor internal structure. Sensors in LIASensor follow a structure that is common to a wide variety of components. However, this was not a liability; on the contrary it allowed the fast development of the prototype for this work. The next chapter approach the subject in a deeply way.

Chapter 4

Implementation

Liasensor is implemented over Castalia[1], a realistic wireless and radio modeling simulator for Wireless Sensor Networks based in the OMNET++ platform and developed in C++. Moreover, Castalia provides extended sensing modeling provisions, such as highly flexible physical process model, and the ability to sensing device noise, bias, and power consumption.

LIASensor is an application layer protocol, which runs over Routing and MAC protocols. It is written in C++ language and works as an independent module, i.e. it does not depend of any other protocol.

This chapter is divided as follows: The first section presents how locality and interest awareness is enforced. The second section, introduces the concept of consistency view in LIASensor. Then in chapter 4.3 it is presented the structure of messages in LIASensor. At last, the section 4.4 explains the structure of LIASensor.

4.1 Locality and interest awareness Enforcement

As section 3.3 presented, locality-awareness uses radius of interests as its main element. The analysis of radius of interest encloses sensors within a consistency degree, which states the response's size.

Every query has a RSSI associated to it, which is used to calculate the distance between the user and the sensor (section 3.7). When a query is transmitted, the nodes that sensed the query read the query's RSSI and compare it with their consistency view, a table that associates RSSI values into consistency views. Consistency views are detailed in section 4.2. If the RSSI matches one

[1] http://castalia.research.nicta.com.au/index.php/en

of the table's rows, the response is constructed and sent with the correct amount of information; otherwise the query is ignored and dropped.

The enforcement of interest-awareness is easily accomplished. The strict comparison between the request type and the sensor skill is sufficient. Every sensor node is skilled to perform some task and every user request has a field that identifies the information on which the user is interested. So, when a query arrives, the sensor node captures the packet, read the type and compares it with its own function. If the type field match its function a new response is sent, otherwise the request is ignored and dropped.

4.2 Consistency View

Sensor nodes have a consistency view, which can be described as a table that contains the interval of RSSI values mapped into consistency degrees. Each consistency view row is composed by: a) a RSSI interval; b) a consistency degree; and c) the amount of information a sensor must forward when a query arrives. The amount of information forwarded should increase as the consistency degree increases too. In other words, higher consistency degrees mean more information.

Values within the consistency table might be replicated without affecting the way consistency views work. When a query arrives, sensors search for matches in this table view through the comparison between the queries' RSSI, and the RSSI interval in the consistency view. The first matched row is returned, and all the others discarded. A response packet is created according with the returned row and forward back to the WSN.

Consistency degree Id	RSSI Interval	Info Quantity
1	]50db, 60db]	Last 1 hour
2	]60db, 70db]	Last 1.5 hours
3	]70db, 80db]	Last 2 hours
4	]80db, 90db]	Last 3 hours
5	]90db, 100db]	Last 4 hours

Table 4.3: Default Consistency View.

Consistency views are a core resource in LIASensor, since it is the only resource that maps a query request to the response's size. By default, sensor nodes are deployed with a general consistency view, which is represented in table 4.3. The default table has five RSSI intervals mapped into five consistency degrees. This table is not editable or erasable; however, the end-user can create and use, as he needs, new consistency views.

The user can opt to create new consistency views, where he can add rows, remove rows, change RSSI values for singular or group of sensors, or even disables the consistency view. These operations can be performed at any given time, over any consistency views that have been created by a user.

The default consistency table was built with these values by a matter of simplicity. These values can be changed, without major problems. Moreover, despite the quantity of information used is in elapsed time, the truth is that these numbers might be represented in many other ways. For instance, LIASensor could send the most relevant information, the maximum or minimum peaks, or any other aggregation considered important to the user.

4.2.1 Consistency degree update

As explained before in section 4.2, the consistency degrees can be created and updated anytime; to do so, there are two distinct ways to update the consistency view:

- Through the dissemination of a single packet containing the new consistency view;
- Inside the query. Besides the query itself, the request packet carries the new consistency view.

LIASensor uses the first approach, because it is the most energy-aware option; the second approach though more reliable, since it guarantees that sensors always know the current consistency view, is also the more expensive, since the packet overhead increases. In addition, the second approach increases the time and energy required to read a packet, as well as, it demands sensors to process the new consistency view in every query arrival. Furthermore, increasing the size of the packet would not be consistency with the rest of this work, which has precisely the opposite goal.

So, by reasons of performance LIASensor uses the first approach. This implies the use of a broadcast message containing the configuration message, but since this message is not meant to be constantly used, the broadcast, which can be energy-expensive, won't be very dramatic.

4.3 Packet Structure

Beyond Castalia packets, LIASensor uses and defines three new packets: 1) a request packet; 2) a reply packet; and 3) a configuration packet.

4.3.1 Request packet

The request packet (Figure 4.12) is always sent by an end-user, and it identifies uniquely its sender, as well as, which sensors must respond and which consistency view must be used to construct the response. Below it is shown which fields compose the packet.

- **Type** - Unsigned Char (1 Byte) - Identifies the type of packet, i.e. (query, response or configuration packet). In this packet the value assigned is the constant ”Request”;
- **Group** - Unsigned Char (1 Byte) - Identifies to which group of sensors the packet is addressed to, e.g. sensors skilled to sense temperature, pressure, etc;
- **User Id** - Unsigned Char(1 Bytes) - Identifies uniquely the end-user;
- **Sequence Number** - Unsigned Integer (4 Bytes) - Identifies uniquely the request. This value is incremented unitarily;
- **Consistency view** - Unsigned Char (1 Byte) - Identifies which consistency view must be used to construct the response.

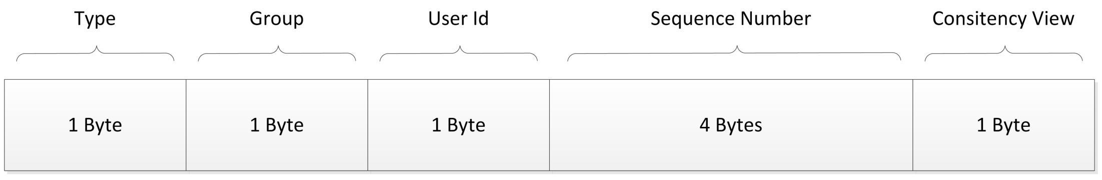


Figure 4.12: Request Packet.

4.3.2 Reply packet

Reply packets (Figure 4.13) are always constructed and sent by sensor nodes. These packets are responses to requests packets, and they contain the environmental data requested, the receiver of the packet, and which consistency view has been used to construct the packet. Below it is shown the packet composition.

- **Type** - Unsigned Char (1 Byte) - Identifies the type of packet, i.e. (query, response or configuration packet). In this packet the value assigned is the constant ”Reply”;
- **Message-group** - Unsigned Char (1 Byte) - Identifies which kind of information the packet contains. For instance, if it is values of temperature, pressure, etc.

- **User Id** - Unsigned Char(1 Byte) - Associates the response with the user who requests the response. This is the request's user id;
- **Response Id** - Unsigned Integer (4 Bytes) - Associates the response with the query. This value is the request's sequence number;
- **Data collected** - Data Object - (Maximum of 2KB) Contains the data collected. This is a c++ object (chapter 4.4.3);
- **Consistency degree** - Unsigned Char - (1 Byte) - Indicates which consistency degree has been used.

The "Response Id" and the "User Id" allows the user to easily filter and drop messages that are not addressed to him. The "Consistency degree" and the "Message-group" allow the user's device to easily process the received information.

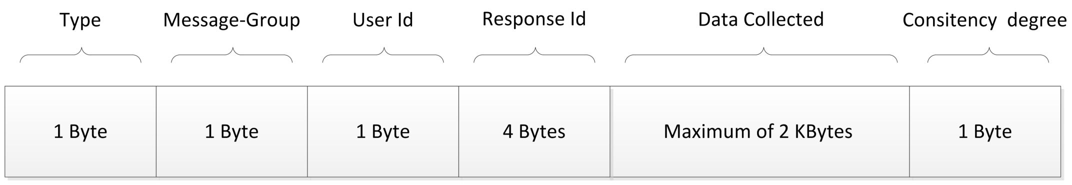


Figure 4.13: Reply Packet.

4.3.3 Configuration packet

Configuration packets (Figure 4.14) are created by end-users and their purpose is configuring consistency views after the WSN deployment. The packet contains the consistency view on which the configuration will be performed, as well as, the update for that consistency view. Below it is shown the composition of configuration packets.

- **Type** - Unsigned Char (1 Byte) - Identifies the type of packet, i.e. (query, response or configuration packet). In this packet the value assigned is the constant "Config";
- **Group** - Unsigned Char (1 Byte) - Identifies to which group of sensors the packet is addressed to, e.g. sensors skilled to sense temperature, pressure, etc.
- **User Id** - Unsigned Char (1 Byte) - Identifies uniquely the end-user;
- **Consistency-view Id** - Unsigned Short Integer (2 Bytes) - Identifies uniquely the consistency view on which the changes will be performed.

- **Operation** - Unsigned Char (1 Byte) - Identifies the operation that will be performed. For instance, creating a new row, or deleting an existing row;
- **Consistency-view** - Data Object - (Maximum of 1 KB) - Contains the new consistency view or the changes to the current consistency view.

The ”Group” field indicates on which sensors the operation will be performed. This field might have values such as ”TEMPERATURE”, ”PRESSURE”, or any other value that represents the sensor skill. Nevertheless, the ”Group” field can be set to a blank value (””), which means that the configuration packet will be propagated throughout the WSN without any restrictions. The ”Operation” field can be set to five different values. It indicates which type of operation will be performed, and it is up to the ”Consistency-view” field to carry the value of the operation.

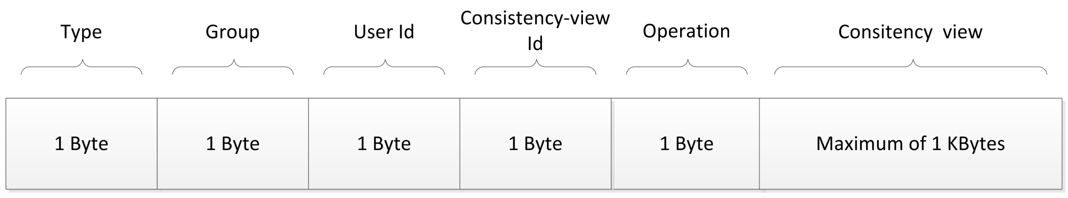


Figure 4.14: Configuration Packet.

The ”Operation” field can have one of these six values:

- Add a new consistency view;
- Add a new row;
- Update an existent row;
- Remove an existent row;
- Delete a consistency view;
- Disable all consistency views.

The ”Consistency-view” is filled differently regarding the ”Operation” field. For instance, if the ”Operation” is set to a ”New row”, the ”Consistency-view” will carry the values of the new row; if is set to ”New consistency view”, the ”Consistency-view” will carry the values of a full consistency view. In other words, the value of the ”Consistency-view” will always be dependent of the chosen ”Operation”.

The consistency view can be updated anytime the user needs to, and depending of the update, the packet parameters vary. Table 4.4 shows which parameters are added to the configuration packet in order to perform each operation.

Type of Operation	Consistency-view Value
Add a consistency view	1 - The consistency view Id; 2 - The consistency view itself.
Add a row	1 - The consistency view Id; 2 - The new row.
Update a row	1 - The consistency view Id; 2 - The row to be updated; 3 - The new value.
Remove a row	1 - The consistency view Id; 2 - The row.
Delete a consistency view	The consistency view Id.
Disable a consistency view	--

Table 4.4: Permited configurations.

In addition to the fields mentioned above, in every packet sent, Castalia automatically includes the Received Signal Strength Indication (RSSI), and the Link Quality Indication (LQI).

4.4 Structure

LIASensor is structured in the same way that Castalia is. The implementation code is organized into modules, and messages and files follow Castalia structure and convections. This structure allows LIASensor to be executed and to take the maximum advantage of Castalia benefits.

4.4.1 Modules

The protocol stack in Castalia and LIASensor is constructed through different and independent modules. In addition, each protocol layer is contained within a sub module. Modules and sub modules are organized in a folder system. In other words, different modules are in different folders, and protocols are contained in subfolders inside the correspondent module.

As figure 3.6 shows, Castalia is divided into three major modules: Application, Routing and MAC modules, which represent respectively the application, transport and network layers. The modules are connected with OMNeT++ NED language.

The NED language is a configuration language, which easily permits the definition of modules i.e. the configuration of names, parameters, and interfaces (gates in and gates out). The decision of which protocols are used within each layer is also made in the .ned file. Moreover, the NED language allows the user to set the number of sensor, as well as, their position and behavior.

Castalia also provides a Resource Manager which keeps track of the energy spent by the node and also holds some node-specific quantities such as the clock drift and the baseline power consumption (minimum power that the node consumes).

Modules that model hardware devices (i.e. the radio and the sensor manager) send messages to the Resource manager in order to signal how much power they currently draw. The resource manager has a complete view of the total power drawn and based on this, it calculates the amount of energy consumed each time a change in power is detected or periodically (if a power change has not happened for some time).

LIASensor is an application module served by Routing and MAC protocols. The transport layer in LIASensor executes a one-hop protocol, whereas the network layer protocol executes a CSMA protocol. These two protocols are very simple and they have been chosen because Castalia does not provide a wide variety of protocols. Nevertheless, the type and complexity of these protocols are not a cause of concern because from the beginning these two protocols were out of the scope of this work, and further, LIASensor evaluation focuses only in the application protocol.

LIASensor is contained in a folder called ”LIASensor”, inside the folder ”Application layer module”. In addition to the components described in the previous chapter, in figure 3.6, we developed a Counting Module for LIASensor, which serves to count the number and size of the received messages. This module was conceived with the purpose of improving and complementing Castalia Resource Manager, which has some limitations in messages counting. This module can be easily turned off and it is only used during the evaluation of LIASensor.

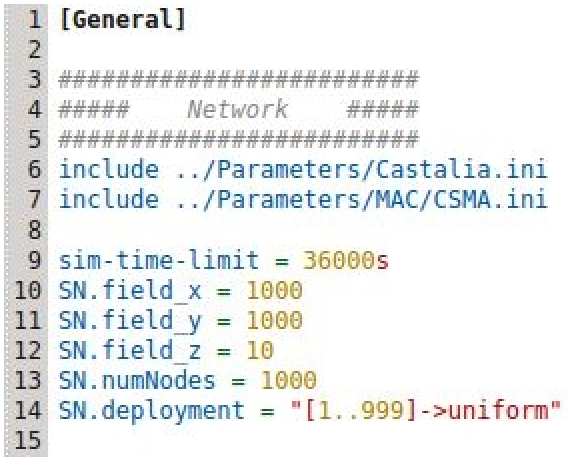
4.4.2 Files

LIASensor is divided into two different components, the configuration file, one NED file (chapter 4.4.1) and the implementation files, the c++ files. The NED file is a single file that contains the network configurations. The implementation files contain the source code of LIASensor and they are compound by a set of .cc files and .h files. Among these files, the core files are LIASensor.h and LIASensor.cc.

4.4.2.1 Ned Files

The NED file is the most important file in a simulation. This file contains the network configuration, as well as, the protocols that will be used. The initial parameters are all set in this file. The LIASensor NED file is called LIASensor.ned.

Figure 4.15 shows the network configuration of LIASensor. The lines 6 and 7 include Castalia and MAC general parameters. Line 9 sets the simulation time, in seconds. Line 10, 11 and 12 sets the size of the region where the WSN will be deployed. At last, line 13 and 14 sets the number of nodes and their distribution.



```
 1 [General]
 2 
 3 #########################
 4 #####    Network    #####
 5 #########################
 6 include ../Parameters/Castalia.ini
 7 include ../Parameters/MAC/CSMA.ini
 8 
 9 sim-time-limit = 36000s
10 SN.field_x = 1000
11 SN.field_y = 1000
12 SN.field_z = 10
13 SN.numNodes = 1000
14 SN.deployment = "[1..999]->uniform"
15 
```

Figure 4.15: LIASensor configuration file.

4.4.2.2 Implementation Files

The main header file, the LIASensor.h, declares the methods and the new c++ class that implements the LIASensor application module, whereas the LIASensor.cc file includes the LIASensor.h and implements the methods that are declared in it.

LIASensor.cc implements almost all the logic in this work; it holds the finite state machine, the client-server architecture, the logic that reads and creates packets, the data objects that store the consistency views, and the enforcement of locality and interest awareness. Besides LIASensor.cc being the core file, there are other files. The solution developed contains objects that save information, namely the consistency view, and the rows that compose the consistency view.

4.4.3 Data Structures

During the execution of LIASensor, sensor nodes need to store some data in order to guarantee LIASensor functionality. These data is saved within the following data structures:

- **SensedData** - A structure that stores all the environmental data sensed;
- **Consistency view** - A structure that store the default consistency view;
- **UserConsistencyView** - A structure that store consistency views of each user.

The ”SensedData” structure is a queue of integers. The queue has been chosen because it operates in a FIFO context (First-in first-out), where elements are inserted into one end of the container and extracted from the other. Thus, elements are organized on a time basis. Recent elements are stored in the front of the queue while old elements are stored in the back of the queue.

The ”ConsistencyView” is an instance of an object called ”ConsistencyView”, which is composed by a set of ”Row” objects.

The ”ConsistencyView” object is a c++ vector. The vector has been chosen due to the ease of iteration and element fetching. In addition, this object is compound by the following variables: a) the number of rows; b) the owner of the consistency view (the ”user-id”); c) the consistency id, which together with the ”user id”, identifies uniquely the consistency view; and d) a vector that contains all the rows.

The ”UserConsistencyView” is a c++ map. The key of the map is the ”user id” concatenated with the ”consistencyView id”, which permits the creation of unique keys. This structure does not allow repeated keys and the element mapped to this key is a ”ConsistenctyView” object.

The ”Row” object represents a single row of the consistency view and is compound by:

- **MinValue** - Unsigned Char (1 Byte) - The minimum value of the RSSI interval;
- **MaxValue** - Unsigned Char (1 Byte) - The maximum value of the RSSI interval;
- **ConsistencyDegree** - Unsigned Char (1 Byte) - The consistency degree label.
- **InfoAmount** - Unsigned Integer (2 Bytes) - The amount of information that should be sent when a query arrives if its RSSI matches this row.

Every row represents a consistency degree. It is considered that a node is within a consistency view if the query received has a RSSI above the ”MinValue” and below the ”MaxValue”.

4.4.4 Messages

As chapter 4.3 presented, LIASensor has three types of data packets: request, reply and configuration packets. These three packets defined by LIASensor inherit from ”ApplicationPacket”, which provides access to Castalia methods and allow them for being used into new application modules. Each one of these packets contains different attributes and methods. In addition, they can only be used in application layer protocols.

Chapter 5

Evaluation

This chapter describes LIASensor's results and evaluation. In order to evaluate LIASensor we made a few simulations and the main goal was the measurement of both network lifetime and energy consumption. As mentioned above (chapter 4) Castalia was the chosen simulator.

In order to compare, in a better and liable way, the simulations have been done in pairs, one without LIASensor and another with LIASensor; the results have been stored, analyzed, and displayed in graphics.

In more detail, this chapter is divided as follows. The first section, presents the simulations details and assumptions, i.e. the conditions behind every simulation, and the network topology. Then, section 5.2, presents the simulation results and analysis.

5.1 Simulations details

In order to compare the obtained results, we performed pairs of simulations; each simulation pair was depicted under similar conditions, with the same number of sensors over the same period of time.

The simulations depicted have 1 (one) sensor node per m^2 (square meter). The environment simulated has 1Km^2 (one square kilometer) of total area. Thus, we simulate an environment with 1000 (one thousand) sensor nodes. The routing and MAC protocols used were existent Castalia protocols. The routing protocol used was a one-hop protocol, i.e. the packets are only delivered to sensors that are at a one-hop distance. On the other hand, the MAC protocol was a simple CSMA protocol.

The simulations were performed during a period equivalent to 10 (ten) real hours, and the sink was able to move throughout the WSN requesting information to sensors. The sink made 300 requests during every simulation, which represents a request in every 2 (two) minutes. The sink used the default consistency view (subsection 4.2), and it did not send any configuration requests.

The only differences between the simulations was the application layer protocol. The first simulation, of each pair, was executed with an existing Castalia's protocol as application layer protocol, whereas the second simulation, of each pair, was executed with LIASensor as application layer protocol.

The wireless sensor network deployed where the simulations were performed had the following topology:

- **Static nodes** - Sensors' position is fixed. They do not move from their initial position;
- **Nondeterministic network** - Sensors are scattered randomly, and the end-user does not know their position;
- **Mobile sink** - The user can move freely through the region;
- **Single sink** - There is a single user gathering information from the environment region;
- **Homogeneous network** - All sensors are power equivalent and they have similar hardware. The only variable is the sensing component, since sensors can have different sensing abilities;

To correctly evaluate LIASensor features, we compare three different common elements of wireless sensor networks. These statistics provide a realistic and reliable source about the network lifetime.

- **Number of messages** - Comparison of the number of messages received, with and without LIASensor. Obtained through LIASensor's Counting Module and Castalia's Resource Manager;
- **Size of message** - Comparison of the messages' size, with and without LIASensor. Obtained through LIASensor's Counting Module and Castalia's Resource Manager;
- **Network lifetime** - Comparison of the network lifetime. We consider that a network is dead when any sensor within the high consistency degree ring cannot respond. Obtained through LIASensor's Counting Module and Castalia's Resource Manager.

These three comparisons allow me to understand, compare and conclude whether or not LIASensor is able to improve WSNs lifetime.

5.2 Results

The results of every simulation were obtained through Castalia's Resource Manager and LIASensor's Counting Module (subsection 4.4.1).

Subsection 5.2.1 presents the results of the main simulation. In this simulation we incorporate all the assumptions made in section 5.1, and the main purpose was the measurement of the lifetime of the network with and without LIASensor, as well as the number and the size of messages received by the sink.

Thus, we made five simulations. The first three tested the impact of locality and interest awareness in the network, first separately and then together. Messages' size and network lifetime were measured in the fourth and fifth simulation respectively.

In subsection 5.2.2, we vary some of the assumptions of section 5.1. So, we decide to vary the number of sensor nodes, the consistency view diameter and the time between query transmissions. The results obtained in these simulations allow understand how these variations impact the lifetime of the whole wireless sensor network.

5.2.1 Main simulation

Figures 5.16 to 5.18 regards the results of simulations performed to count the number of messages received by the sink. The graphs display the number of messages with and without LIASensor. The x-axis shows the number of requests, whereas the y-axis shows the number of messages in units. Each peak represents the number of messages that arrived after a request.

- **Number of messages with Locality Awareness**

The graph in figure 5.16 represents the results of a simulation executed with locality awareness turned on, and interest awareness turned off. In this graph we confirm that locality awareness is capable of reducing the number of messages.

We also detect in average a decrease of 19% in messages' number and a standard deviation of 2.23. This result can be justified with the consistency degrees that were formed throughout the network. As have been previously told, sensors positioned behind the last consistency degree do not respond, which results in a minor number of messages within the network.

- **Number of messages with Interest Awareness**

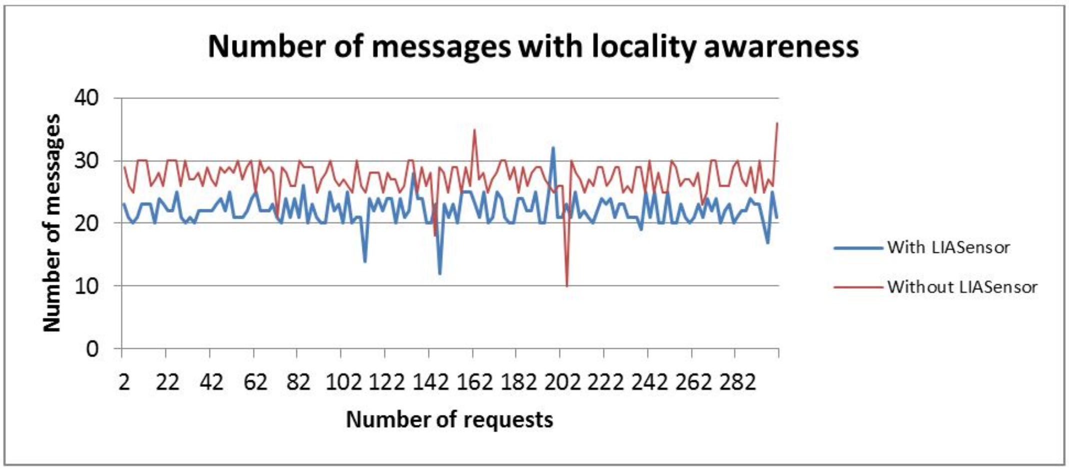


Figure 5.16: Number of messages received in LIASensor with locality awareness.

Figure 5.17 shows the results of the second simulation; in this simulation LIASensor was executed with locality awareness turned off and interest awareness turned on. As expected, the number of messages decreased too. We notice a 37.8% decrease, in average, of messages numbers and a standard deviation of 2.45. These results are only possible, because with interest awareness turned on, only sensors with specifics skills answered the query.

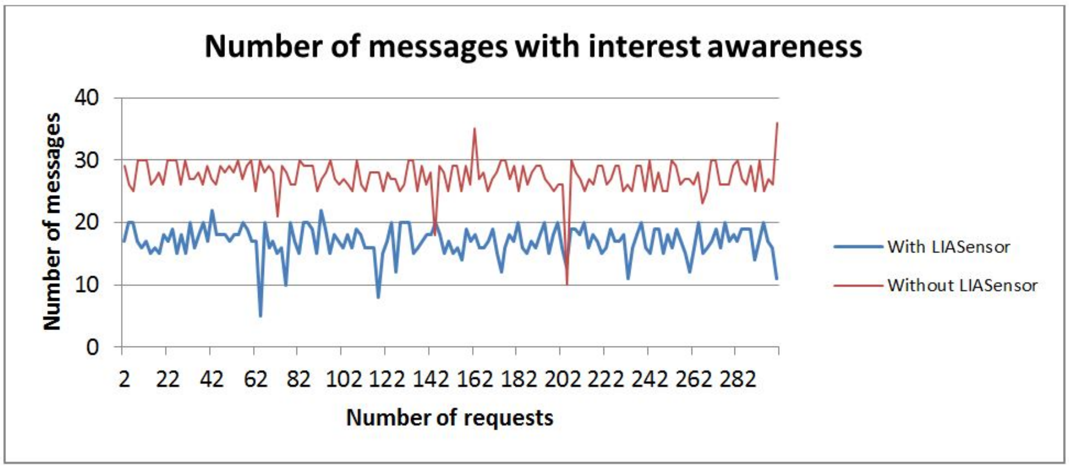


Figure 5.17: Number of messages received in LIASensor with interest awareness.

- **Number of messages with Interest and Locality Awareness**

At last, figure (figure 5.18) presents the result of LIASensor fully operational, with locality and interest awareness turned on. This graph clearly shows the decreasing of messages in LIASensor. As explained before, locality awareness avoids messages from farther sensors, whereas interest awareness avoids useless messages to the user. Thus, as expected, these two techniques combined decreased, even more, the number of messages. With both interest and locality awareness we notice, in average, a decrease of 48% in messages numbers and a standard deviation of 1.78.

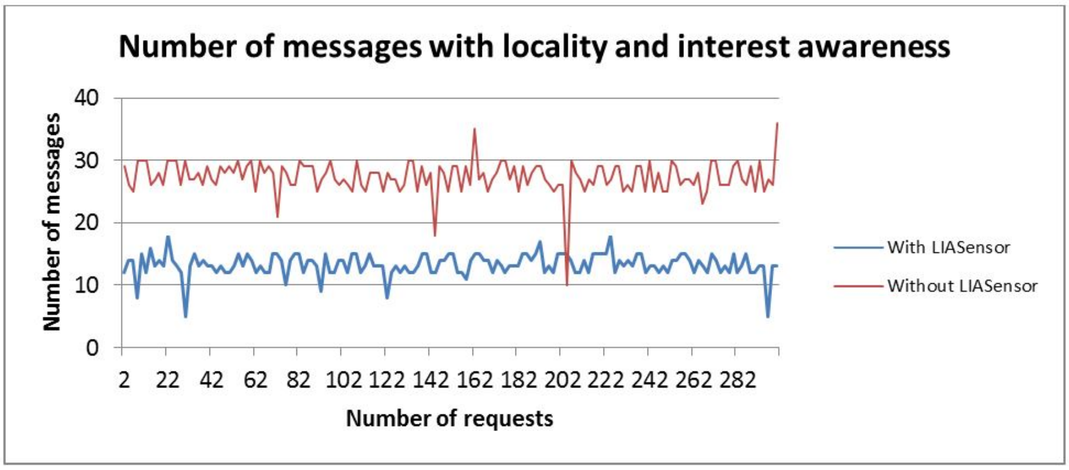


Figure 5.18: Number of messages received in LIASensor with interest and locality awareness.

- **Size of messages**

Figure 5.19 shows the results of the simulation depicted to test the average packet size. The graph represents the average size of a packet containing environmental data. In this graph, just like the previous one, the x-axis represents the number of requests. The y-axis now represents the size of packets in Bytes.

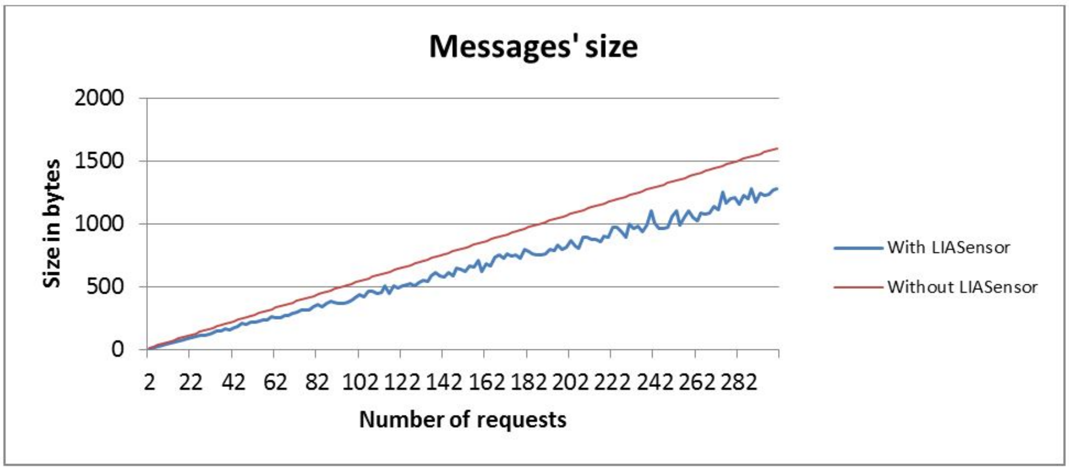


Figure 5.19: Average packet size in LIASensor.

As expected, LIASensor reduces the size of messages within a WSN. Through the graph analyzes, we can infer that LIASensor decreases packets' size some minutes after the network deployment. LIASensor reduced the size of packets in 21.9%. This behavior is explained by the use of locality awareness, since this feature associates the sensor response's size with the sensor distance to the end-user.

- **Network lifetime**

To test the network lifetime, simulations were executed until the death of the network. It was considered that a network was dead when all the sensors within the most internal consistency degree did not respond.

	With LIASensor	**Without LIASensor**
Time in hours	18.26	15.26

Table 5.5: Network lifetime with LIASensor.

Table 5.5 shows the results of this simulation. The obtained results show a 16.4% of improvement. Without LIASensor, the network died after 15.26 hours of execution, whereas with LIASensor, the network only died after 18.26 hours of execution. These results are in accordance with the other results.

As it is known, sensors spent a large amount of their energy when they are sensing and transmitting data, so by reducing the number of messages circulating in the network, as well as, the size of the messages in it, the increase of the network lifetime is a natural result.

5.2.2 Complementary simulation

The following results, aim to measure the lifetime of the WSN with the new parameters. Unlike the main simulation depicted above, the measure of the number and the size of messages exchanged between the sensors and the sink is not a primary goal. The purpose of these simulations is the measurement of the network lifetime. Figures 5.20 to 5.22 presents the results of these simulations and their analysis.

Figure 5.20 presents the variation of consistency degree range. The x-axis of the graph represents the difference between the maximum and the minimum values in a consistency degree; the value is measured in decibel (dB). The y-axis measures the network lifetime in hours.

Figure 5.21 presents the variation of number of sensors. In this graph the x-axis represents the number of sensor nodes in units, whereas the y-axis measures the total number of messages received by the sink.

At last, figure 5.22 presents the variation of the period between each request. The x-axis measures the period between requests in seconds while the y-axis measures the lifetime of the network.

In every graph the square symbol represents the value that was obtained in the main simulation.

- **Consistency degree range**

In the first simulations depicted the values for consistency degrees were the default values (table 4.3). These values can be changed: a) during the simulation (in execution time); and b) at the beginning of the simulation (in compilation time).

In order to imitate the main simulation, we decide to change these values at the beginning of the simulation. Figure 5.20 shows how the network lifetime varies when the consistency degree range also varies.

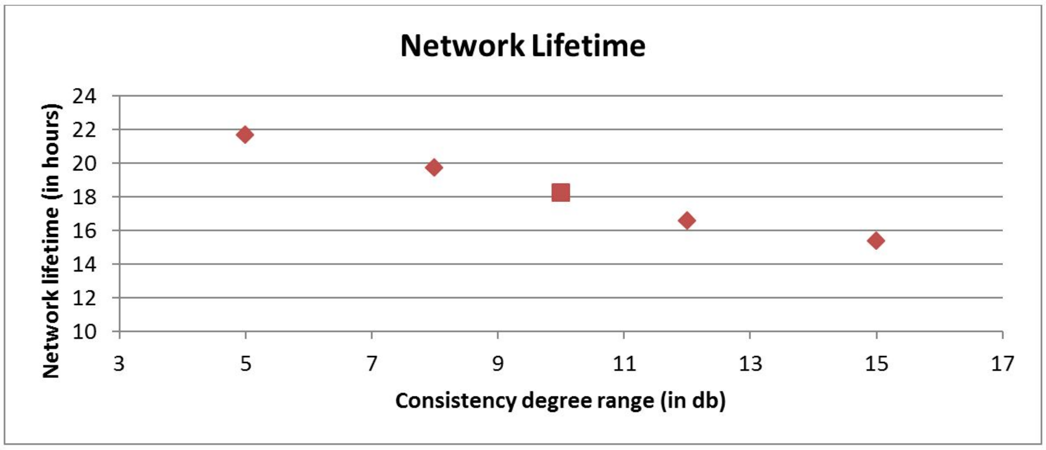


Figure 5.20: Network lifetime with varying consistency degrees.

Naturally, the lifetime of the network decreases as the diameter of the consistency degrees increases. These results are explained by the increasing number of messages transmitted.

As the consistency degrees increases more sensors are seized inside the transmission range, which make then transmit messages. This increases the number of packets transmitted and the power consumed for every sensor, which reduces the lifetime of the entire network. The square in the graph represents the value of the previous simulation (a consistency degree range of 10 results in a 18.26 hours of networking).

- **Sensor number**

In the main simulation, described in section 5.2, we choose 1000 sensors, which correspond to 1 sensor node per 1 m^2 (one square meter). In this simulation we vary the number of sensors from

400 to 2000. Figure 5.21 shows the number of messages received by the sink when the number of sensor varies.

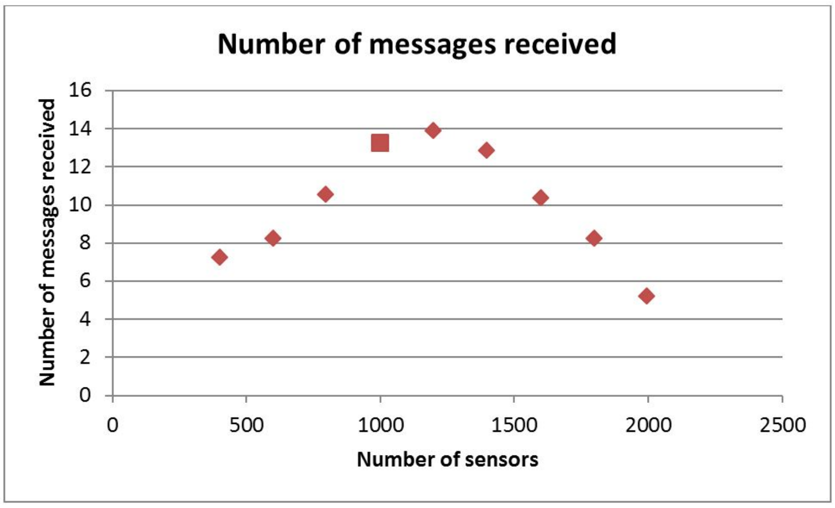


Figure 5.21: Number of messages received with varying number of sensors.

The results in graph 5.21 may be unnatural. Naturally we could think that a major number of sensors would increase the number of packets received. Nevertheless, the explanation is very simply. A larger number of sensors within the same space also increase the number of collisions between messages. In this simulation we note a large amount of collisions between messages, which decreased the amount of legible messages.

- **Transmission interval**

In the first simulations we opt to query the network in every two minutes (120 seconds). However, in this simulation we vary these numbers from 30 seconds to 180 seconds. Figure 5.22 shows the variation of the network lifetime when the time between requests varies.

By analysis of the graph we conclude an increase of the network lifetime when the interval between requests increases. This is a natural result since query transmission is one of the most energy-expensive tasks a sensor can make.

5.3 Summary

This chapter described the details and the results of LIASensor's assessment. It has been shown that LIASensor is capable of decreasing the number of messages, as well as, reducing the size of messages, which naturally resulted in an increase of the network lifetime.

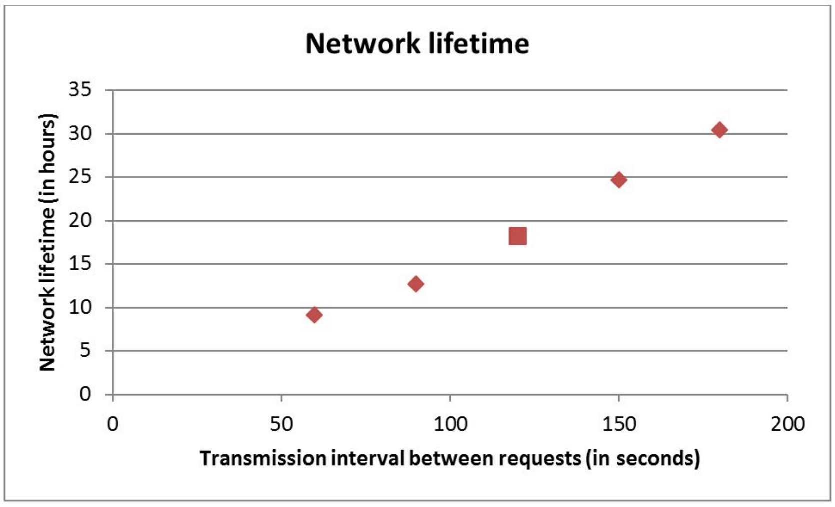


Figure 5.22: Network lifetime with varying transmission time.

All the results obtained were very encouraging. In the main simulation section, the numbers shows that the simulation with LIASensor got better results than the simulation without LIASensor. Moreover, the results proved that interest and locality awareness achieves very good results. In fact, each one of these techniques reduces the number of messages individually.

The explanation for these results can be easily extracted from LIASensor architecture. The architecture achieved to prioritize messages from sensors closer to the user rather than messages from sensors farther. In addition, the reduction of the amount of information within every response also contributed for these good results.

The complementary simulation proves that the performance of results is dependent of many variables, as many other complex systems. It also shows that the main simulation has not set its variables with values that support the best performance and the best results. This situation happened because the perfect result is not a major goal in this work, but rather the proof that LIASensor increases the lifetime of the network.

In addition, the best values for variables in Castalia would not be the best values in other simulators, or even in real networks.

Chapter 6

Conclusions and Future work

This dissertation presented the research work that was conducted in the context of my MSc thesis.

This report described two techniques capable of increasing the lifetime of a Wireless Sensor Network, namely Interest and Locality awareness techniques. It also had shown the implementation details and the results of my work. In order to do this report I studied different methods and algorithms able to increase a network lifetime, I also studied techniques able to filter messages based in their location and interest. Based in the study conducted, I was able to propose LIASensor, an application layer protocol for Wireless Sensor Networks.

To test the results of my work, I chose a virtual simulator to test the application. The simulator, though not real, could guarantees very realistic results at a minimum cost. Moreover, nowadays, these simulators simulate real networks in very genuine and reliable ways, without all the costs of a real network.

Using my solution, I was capable to clearly show a decrease of messages size and number, and consequently an improvement in network lifetime.

6.1 Future work

In my thesis LIASensor has been implemented as a full application layer protocol; however, it could be done as an add-on to other application layer protocols. This would imply some adaptations to LIASensor, since the application protocols would have to communicate with LIASensor before they could make any decision.

Another improvement over LIASensor would be an adaptation to static sinks. LIASensor was developed to work with mobile sinks. With LIASensor the sensors that respond to queries are those which are at a single-hop distance. This new adaptation would require a multi-hop communication between sensors and probably a different routing algorithm. For instance, if the sink is located in the network periphery and the user wants to know information about the other end of the network, the query, and the responses, would have to travel through a multi-hop communication.

As it has been presented, using a GPS component to calculate the distance between the nodes and the sink would also increase the reliability of the system. However, the cost associated to these components does not justify the effort. In the future, if the prices of GPS components came down, the integration of such components would be an asset to the whole system.

At last, would be interesting to do further tests in real networks, in order to know how would be the real performance of LIASensor.

Bibliography

[1] AKYILDIZ, I., SU, W., SANKARASUBRAMANIAM, Y. & CAYIRCI, E. (2002). A survey on sensor networks. *Communications Magazine, IEEE*, **40**, 102 – 114.

[2] AKYILDIZ, I.F., SU, W., SANKARASUBRAMANIAM, Y. & CAYIRCI, E. (2002). Wireless sensor networks: a survey. *Computer Networks*, **38**, 393–422.

[3] AL-KARAKI, J. & KAMAL, A. (2004). Routing techniques in wireless sensor networks: a survey. *Wireless Communications, IEEE*, **11**, 6 – 28.

[4] BRAGINSKY, D. & ESTRIN, D. (2002). Rumor routing algorthim for sensor networks. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, WSNA '02, 22–31, ACM, New York, NY, USA.

[5] CHAURASIYA, S., PAL, T. & BIT, S. (2011). An enhanced energy-efficient protocol with static clustering for wsn. In *Information Networking (ICOIN), 2011 International Conference on*, 58 –63.

[6] CHAURASIYA, S., SEN, J., CHATERJEE, S. & BIT, S. (2012). An energy-balanced lifetime enhancing clustering for wsn (eblec). In *Advanced Communication Technology (ICACT), 2012 14th International Conference on*, 189 –194.

[7] CHEN, B., JAMIESON, K., BALAKRISHNAN, H. & MORRIS, R. (2002). Span: an energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *Wirel. Netw.*, **8**, 481–494.

[8] CLARE, L.P., POTTIE, G.J. & AGRE, J.R. (1999). Self-organizing distributed sensor networks. 229–237.

[9] DONG, M., YUNG, K. & KAISER, W. (1997). Low power signal processing architectures for network microsensors. In *Low Power Electronics and Design, 1997. Proceedings., 1997 International Symposium on*, 173 –177.

[10] FANG YAN, X., HUI GU, X., LI, L. & YOU SONG, J. (2012). A hierarchical clustering algorithm based on energy and distance balancing for wsn. In *Intelligent Computation Technology and Automation (ICICTA), 2012 Fifth International Conference on*, 463 –466.

[11] GODWINPREMI, M. & SHAJI, K. (2010). Router performance with aoi and roi routing for wireless sensor networks. In *Communication Control and Computing Technologies (ICCCCT), 2010 IEEE International Conference on*, 236 –240.

[12] HASBULLAH, H. & NAZIR, B. (2010). Region-based energy-aware cluster (rec) for efficient packet forwarding in wsn. In *Information Technology (ITSim), 2010 International Symposium in*, vol. 3, 1 –6.

[13] HEINZELMAN, W., CHANDRAKASAN, A. & BALAKRISHNAN, H. (2000). Energy-efficient communication protocol for wireless microsensor networks. In *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*, 10 pp. vol.2.

[14] HEINZELMAN, W., CHANDRAKASAN, A. & BALAKRISHNAN, H. (2002). An application-specific protocol architecture for wireless microsensor networks. *Wireless Communications, IEEE Transactions on*, **1**, 660 – 670.

[15] HEINZELMAN, W.R., KULIK, J. & BALAKRISHNAN, H. (1999). Adaptive protocols for information dissemination in wireless sensor networks. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, MobiCom '99, 174–185, ACM, New York, NY, USA.

[16] INTANAGONWIWAT, C., GOVINDAN, R. & ESTRIN, D. (2000). Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, MobiCom '00, 56–67, ACM, New York, NY, USA.

[17] INTANAGONWIWAT, C., GOVINDAN, R., ESTRIN, D., HEIDEMANN, J. & SILVA, F. (2003). Directed diffusion for wireless sensor networking. *Networking, IEEE/ACM Transactions on*, **11**, 2 – 16.

[18] KATIYAR, V., CHAND, N., GAUTAM, G. & KUMAR, A. (2011). Improvement in leach protocol for large-scale wireless sensor networks. In *Emerging Trends in Electrical and Computer Technology (ICETECT), 2011 International Conference on*, 1070 –1075.

[19] KRISHNAMACHARI, L., ESTRIN, D. & WICKER, S. (2002). The impact of data aggregation in wireless sensor networks. In *Distributed Computing Systems Workshops, 2002. Proceedings. 22nd International Conference on*, 575 – 578.

[20] KULIK, J., HEINZELMAN, W. & BALAKRISHNAN, H. (2002). Negotiation-based protocols for disseminating information in wireless sensor networks. *Wirel. Netw.*, **8**, 169–185.

[21] LI, L. & BAI, F. (2011). Analysis of data fusion in wireless sensor networks. In *Electronics, Communications and Control (ICECC), 2011 International Conference on*, 2547 –2549.

[22] LIANG, Y. & YU, H. (2005). Energy adaptive cluster-head selection for wireless sensor networks. In *Parallel and Distributed Computing, Applications and Technologies, 2005. PDCAT 2005. Sixth International Conference on*, 634 – 638.

[23] LINDSEY, S. & RAGHAVENDRA, C. (2002). Pegasis: Power-efficient gathering in sensor information systems. In *Aerospace Conference Proceedings, 2002. IEEE*, vol. 3, 3–1125 – 3–1130 vol.3.

[24] LINDSEY, S., RAGHAVENDRA, C. & SIVALINGAM, K. (2001). Data gathering in sensor networks using the energy*delay metric. In *Parallel and Distributed Processing Symposium., Proceedings 15th International*, 2001 –2008.

[25] LIU, Y., GAO, J., ZHU, L. & ZHANG, Y. (2009). A clustering algorithm based on communication facility in wsn. In *Communications and Mobile Computing, 2009. CMC '09. WRI International Conference on*, vol. 2, 76 –80.

[26] LOSCRI, V., MORABITO, G. & MARANO, S. (2005). A two-levels hierarchy for low-energy adaptive clustering hierarchy (tl-leach). In *Vehicular Technology Conference, 2005. VTC-2005-Fall. 2005 IEEE 62nd*, vol. 3, 1809 – 1813.

[27] MANJESHWAR, A. & AGRAWAL, D. (2001). Teen: a routing protocol for enhanced efficiency in wireless sensor networks. In *Parallel and Distributed Processing Symposium., Proceedings 15th International*, 2009 –2015.

[28] MANJESHWAR, A. & AGRAWAL, D. (2002). Apteen: a hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks. In *Parallel and Distributed Processing Symposium., Proceedings International, IPDPS 2002, Abstracts and CD-ROM*, 195 –202.

[29] MODARES, H., SALLEH, R. & MORAVEJOSHARIEH, A. (2011). Overview of security issues in wireless sensor networks. In *Computational Intelligence, Modelling and Simulation (CIMSiM), 2011 Third International Conference on*, 308 –311.

[30] MORSE, K.L. (1996). Interest management in large-scale distributed simulations.

[31] PARADISO, J. & STARNER, T. (2005). Energy scavenging for mobile and wireless electronics. *Pervasive Computing, IEEE*, **4**, 18 – 27.

[32] PATWARI, N., ASH, J., KYPEROUNTAS, S., HERO, A., MOSES, R. & CORREAL, N. (2005). Locating the nodes: cooperative localization in wireless sensor networks. *Signal Processing Magazine, IEEE*, **22**, 54–69.

[33] POTTIE, G.J. & KAISER, W.J. (2000). Wireless integrated network sensors. *Commun. ACM*, **43**, 51–58.

[34] REN, P., QIAN, J., LI, L., ZHAO, Z. & LI, X. (2010). Unequal clustering scheme based leach for wireless sensor networks. In *Genetic and Evolutionary Computing (ICGEC), 2010 Fourth International Conference on*, 90 –93.

[35] SADAGOPAN, N., KRISHNAMACHARI, B. & HELMY, A. (2003). The acquire mechanism for efficient querying in sensor networks. In *Sensor Network Protocols and Applications, 2003. Proceedings of the First IEEE. 2003 IEEE International Workshop on*, 149 – 155.

[36] SANTOS, M., VEIGA, L. & FERREIRA, P. (2011). Vfc-reckon for android mobile devices.

[37] SANTOS, N., VEIGA, L. & FERREIRA, P. (2007). Vector-field consistency for ad-hoc gaming. In *Proceedings of the 8th ACM/IFIP/USENIX international conference on Middleware*, MIDDLEWARE2007, 80–100, Springer-Verlag, Berlin, Heidelberg.

[38] TARIGH, H. & SABAEI, M. (2011). A new clustering method to prolong the lifetime of wsn. In *Computer Research and Development (ICCRD), 2011 3rd International Conference on*, vol. 1, 143 –148.

[39] VLAJIC, N. & XIA, D. (2006). Wireless sensor networks: to cluster or not to cluster? In *World of Wireless, Mobile and Multimedia Networks, 2006. WoWMoM 2006. International Symposium on a*, 9 pp. –268.

[40] XIA, D. & VLAJIC, N. (2006). Near-optimal node clustering in wireless sensor networks for environment monitoring. In *Electrical and Computer Engineering, 2006. CCECE '06. Canadian Conference on*, 1825 –1829.

[41] XIANGNING, F. & YULIN, S. (2007). Improvement on leach protocol of wireless sensor network. In *Sensor Technologies and Applications, 2007. SensorComm 2007. International Conference on*, 260 –264.

[42] XU, Y., HEIDEMANN, J. & ESTRIN, D. (2001). Geography-informed energy conservation for ad hoc routing. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, MobiCom '01, 70–84, ACM, New York, NY, USA.

[43] YAO, Y. & GEHRKE, J. (2002). The cougar approach to in-network query processing in sensor networks. *SIGMOD Rec.*, **31**, 9–18.

[44] YASSEIN, M.B., AL-ZOU'BI, A., KHAMAYSEH, Y. & MARDINI, W. (2009). Improvement on leach protocol of wireless sensor network (vleach). *JDCTA*, **3**, 132–136.

[45] YOUNIS, O. & FAHMY, S. (2004). Heed: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. *Mobile Computing, IEEE Transactions on*, **3**, 366 – 379.

[46] YU, Y., GOVINDAN, R. & ESTRIN, D. (2001). Geographical and energy aware routing: a recursive data dissemination protocol for wireless sensor networks.

[47] ZAHMATI, A.S., ABOLHASSANI, B., ASGHAR, A., SHIRAZI, B. & BAKHTIARI, A.S. (2007). An energy-efficient protocol with static clustering for wireless sensor networks.

[48] ZHENG, J. & JAMALIPOUR, A. (2009). *Wireless Sensor Networks: A Networking Perspective*. Wiley.