DOKUZ EYLÜL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

WIRELESS SENSOR NETWORKS, PROTOCOLS AND APPLICATIONS

by

Alaauldin IBRAHIM

August, 2011

İZMİR

WIRELESS SENSOR NETWORKS, PROTOCOLS AND APPLICATIONS

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University In Partial Fulfillment of the Requirements for the Degree of Master of Science in Structural Engineering

by
Alaauldin IBRAHIM

August, 2011
İZMİR

M.Sc THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"WIRELESS SENSOR NETWORKS, PROTOCOLS AND APPLICATIONS"** completed by **ALAAULDIN IBRAHIM** under supervision of **ASST.PROF.DR. ŞEN CAKIR** and we certify that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Asst.Prof.Dr. Şen ÇAKIR

Supervisor

Prof. Dr. Cüneyt Güzeliş

(Jury Member)

Dr. Malik Kemal Şiş

(Jury Member)

Prof.Dr. Mustafa SABUNCU

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGEMENTS

I would like to express my great admiration and send my special thanks to my supervisor Yrd. Doç. Dr Şen ÇAKIR and Öğr. Göv. Dr. Malik Kemal ŞİŞ for their precious patience, guidance and effort of teaching desire throughout my thesis and gradation. This thesis would not be completed without their supports.

I also would like to thank my Friends Phd. Student Mustafa AL-HARBAWI for his assistance and Burçin YAŞAR for her Moral and English supports.

Finally, I give my special thanks to my family especially to my mother, Father, sisters and brothers and to all whom always being supporting me, for all their love and wonderful support throughout my master degree.

ALAAULDIN IBRAHIM

WIRELESS SENSOR NETWORK, ROTOCOLS AND APPLICATIONS

ABSTRACT

Wireless sensor network (WSN) is one of the emerging and fast growing fields in the scientific world. However, the major issue of sensor network nodes is the limited energy, which is normally battery operated. This has posed additional challenges to the communication protocols. Hence, most of the attention has been given to the routing protocols.

In this thesis, after mentioning to the general information of WSN, hardware and network architecture of sensor nodes, applications of WSNs, clustering and routing protocols, and making a deep and integrated comparison between some well-known energy efficient routing protocols, a new routing protocol has been proposed called **T**wo **W**ay **E**fficient **L**ocation-based **G**ossiping protocol (TWELGossiping).

The simulation results shows that the TWELGossiping has addressed some drawbacks of counterpart routing protocols, like end to end delay, high packet lost and high energy consumption of the network overall. Moreover, in this work extra equipment/hardware, like Global Position System (GPS) has not used. Hence, cost of the network overall has been reduced.

Keywords: WSNs, WSNs applications, clustering and routing protocols, TWELGossiping routing protocol.

KABLOSUZ SENSÖR AĞLARI, PROTOKOLLER VE UYGULAMALARI

ÖZ

Kablosuz Sensör Ağları (KSA), bilim dünyasında ortaya çıkan ve hızlı büyüyen alanlardan biridir. Ancak, en önemli problem Kablosuz Sensör Ağlarının duğumleri pille çalıştırıldıklarından kısıtlı enerji olanaklarıdır. Bu durum, iletişim protokolleri için ek zorluklar yaratmıştır. Bu nedenle ilginin çoğu yönlendirme protokollerine verilmiştir.

Bu tezde, KSA genel bilgi, donanım ve sensör düğümlerinin ağ mimarisi, KSA uygulamaları, kümeleme ve yönlendirme protokollerine deyinilmiş ve bazı iyi bilinen enerji verimli yönlendirme protokolleri arasında derin ve entegre bir karşılaştırma yapıldiktan sonra Çift Yönlü Verimli Lokasyon bazlı Dedikodu (ÇYVLDedikodu) protokolü adlı yeni bir yönlendirme protokolü önerilmiştir.

Simülasyon sonuçları, Çift Yönlü Verimli Lokasyon bazlı Dedikodu ÇYVLDedikodu protokolü, karşılaştırılan yönlendirme protokollerinin uçtan uca gecikme, yüksek paket kaybı ve ağ genelinde yüksek enerji tüketimi gibi dezavantajlarını gidermiştir. Ayrıca, bu çalışmada Küresel Konumlama Sistemi (KKS) gibi fazladan bir ekipman kullanılmamıştır. Sonuç olarak da, genel ağ maliyetini duşurulmuştur.

Anahtar sözcükler: KSA, KSA uygulamaları, kümeleme ve yönlendirme protokolleri, ÇYVLDedikodu yönlendirme protokolu.

CONTENTS

Page

CHAPTER ONE
INTRODUCTION

The dramatic increase in sensors application over the past 20 years, made it clear that the sensors will make a revolution like that witnessed in microcomputers in the 1980s. Moreover, the first decade of the 21st century has been labeled by some as the *“Sensor Decade”* (Wilson, 2005).

Many advanced have been made in sensor technologies and which are as varied as the applications and many more are in progress. It has been reasonable to design and develop small size sensor nodes of low-cost and low-power which can communicate over an RF (Radio Frequency) channel independently and in short distance (Ilyas & Mahgoub, 2005), especially after the recent advances in micro electro-mechanical systems (MEMS) technology, digital electronics and wireless communications. Increasing the capabilities of these small sensor nodes like sensing, data processing and communicating enable realization of wireless sensor networks (WSNs).

The close interaction of WSNs with the physical world by providing real-time information, furthermore, the distributed sensing capability and the ease of the deployment of sensor nodes make WSNs an important component and integral part of our life. Most or all of WSNs are designed to the only requirements of certain sensing and monitoring applications (Akyildiz & Canvuran, 2010).

WSNs consist of tiny sensor nodes that, in turn, consist of sensors (temperature; light; humidity; radiation; and more), microprocessor, memory, transceiver, and power supply. In order to realize the existing and potential application for WSNs advanced and extremely efficient communication protocols are required. Sensor nodes, in addition to sending the observed data, can make some computations and accomplish complex tasks by using their processing capability. Moreover, they can transmit only the required processed data not raw of data.

Sensor nodes are powered by limited capacity of batteries. Because of the power management activities of these sensor nodes, the network topology is dynamically changes even if the sensor nodes are stationary not mobile (Akyildiz & Canvuran, 2010). These essential properties pose additional challenges to the communication protocols.

WSNs differ from other networks, like internet, where they are often application specific designed and deployed for special purposes (Swami, Zhao, Hong, & Tong, 2007). Opposite the traditional networks which are designed to improve throughput, delay and other performance metric, WSN protocols primarily focus on power conservation. For example, when designing the WSN applications and communication protocols it must provide high energy efficiency. Another factor that must be taken into consideration when designing WSN protocols is the deployment. In some applications, the sensor nodes are randomly deployed need not to be engineered and this deployment requires self-organizing protocols for the communication protocol stack. Moreover, the short transmission ranges pose large numbers of sensor nodes to be deployed very close to each other. Hence, instead of traditional single hop communication that consumes high power, a multi-hop communication is used between these nodes and this leads to less power consumption.

WSNs have wide range of applications as diverse as sensors (temperature, humidity, pressure, noise, light, seismic, thermal, visual, infrared, and more), which are capable of monitoring a wide variety of physical conditions (Raghunathan, Schurgers, Park, & Srivastava, 2002). Wireless Sensor Networks in simple form and according to (Akyildiz, Su, Sankarasubramaniam, & Cayirci, 2002) can be defined as:

A wireless sensor network (WSN) in its simplest form can be defined as a network of (possibly low-size and low-complex) devices denoted as *nodes* that can sense the environment and communicate the information gathered from the monitored field (e.g., an area or volume) through wireless links; the data is forwarded, possibly via multiple hops relaying, to a *sink* (sometimes denoted as *controller* or *monitor*) that can use it locally, or is connected to other networks (e.g., the Internet) through a gateway. The nodes can be stationary or moving. They can be aware of their location or not. They can be homogeneous or not (Verdone, R., et al, 2007, p. 1). Figure 1.1 illustrates this concept in best way.

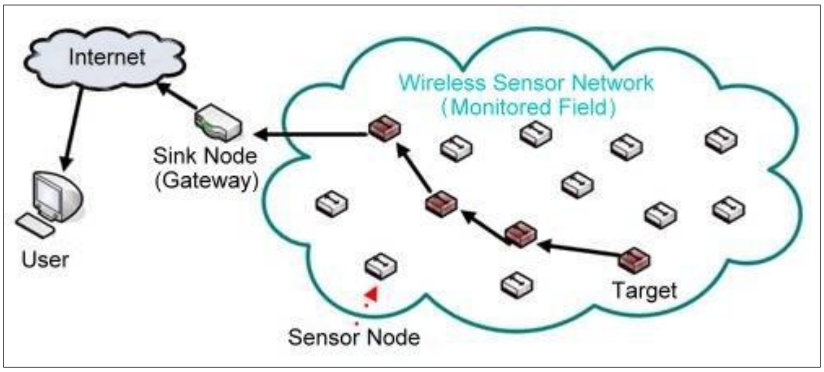

Figure 1.1 Traditional WSN

According to architectural consideration, WSNs can be classified into two types (Chakrabarti & Seberry, 2006): first, Hierarchical Wireless Sensor Networks (HWSN), and second Distributed Wireless Sensor Networks (DWSN).

- ***Hierarchical Wireless Sensor Networks (HWSN):*** This type is more suitable for application that network topology is known before deployment. The nodes in this type, according to their capabilities, are classified into three types; Base stations, Cluster heads, and Sensor nodes Sensor nodes sense the environment and send the sensed data to cluster heads which in turn foreword it to the base station. Data in WSNs flows in three modes:

    - *Unicast* (Sensor to sensor) and sometimes called local communication: we can see this type when sensor nodes send message to other sensor nodes to discover and coordinate with each other.
    - *Multicast* (group wise): when a base station sends a query to some nodes or even when sensor nodes broadcast a message to neighbor nodes for the purpose of what.

- *Broadcast* (base station to sensor): this type occurs when a base station sends a query or control information to the sensor nodes.

- ***Distributed Wireless Sensor Networks (DWSN):*** Network topology is unknown and there is no fixed infrastructure. As soon as the nodes are spread, they start scanning their coverage area and find their neighbor nodes. Data flows in the same way that flows in HWSN except that the broadcast may be happen between any two nodes.

1.1 Advantages of WSN

Reliability, accuracy, flexibility, ease of deployment and cost effectiveness are the basic purposes of Wireless Sensor Networks (WSNs). Wireless Sensor Networks have many characteristics and benefits, so, nowadays you can see it approximately in every field from indoor to outdoor using. Below some of these benefits (Pathan, Hong, & Hyung, 2006):

***Fault Tolerance*:** In individual sensors, this benefit can be achieved by device and information redundancy.

Operability in Harsh Environments: Sensor nodes, due to high level of fault tolerance can be deployed in hard in harsh environments, for example, fire monitoring in forces, in places not easy to reach and in volcano monitoring. This makes WSNs more effective.

Area Coverage: A huge number of sensor nodes can be deployed to monitor a physical environment, for example agricultural applications. Moreover, this number is expendable to involve more sensor nodes and without impacting the network cost.

Connectivity*:* WSNs consist of sensor nodes and sinks, and some times, a gateway. This gateway can be a gate to interconnect with other networks like internet. To make control of WSNs easiest, clustering is used. Each cluster focus on a specific event and these clusters (individual networks) can share their relevant information easily.

Sensing Accuracy**:** The information gathered by large number and different types of sensor nodes is more accurate than the information gathered by a single sensor or few number of sensor nodes.

Minimal Human Interaction: As mentioned above, in some applications, sensor nodes have to be deployed in harsh environment and in places difficult to reach. Hence, this leads to minimum interruption of the WSNs by human.

Dynamic Sensor Scheduling*:* According to the used application, WSNs able to set priority for transmitted data by implying some scheduling scheme.

Wireless Communication: Although building sensor network using existing wired network for some application scenarios is easy, for many application types wired sensor network constructs a big obstacle. Furthermore, the prime advantage of sensor network it is being wireless. Normally sensor nodes are communicated wirelessly through RF channel. The main reasons for sensor network to be wireless, are listed below:

- High cost of wiring (US$20 – US$2000 per foot). In addition to the cost of the other devices that are used in wired networking technology.
- Wiring spends 20% - 80% of installation time and disrupts normal business operations.
- With wired networks there is no redundancy only one path.
- For large number of device, wires constitute a maintenance problem and it will be difficult to reach locations.
- Wires prevent nodes from being mobile.
- Wires maybe prevent sensors from being close to the phenomenon that they control. Then wireless communication between these sensors an inevitable requirement.
- Keep the mess of wiring you see in figure 1.2 on side!

Figure 1.2 Mess of wiring

1.2 Factors Influencing WSN Design

Designing of WSNs requires an extensive knowledge of networking, wireless communication, digital signal processing, embedded systems and software engineering. Therefore, many factors that influencing the design are addressed by the researchers, like hardware constraints, fault tolerance, scalability, production costs, sensor network topology, transmission media; and power consumption (Akyildiz & Canvuran, 2010). Below are explanations for each one:

- ***Hardware Constraints:*** Basic components and general architecture of wireless sensor device (described in detail in chapter two) are shown in the figure 1.3. It consists of these five basic units:

1. Sensing unit
2. Communication unit
3. Processing unit
4. Memory unit
5. Power unit

Moreover, according to the application, additional components could be merged into the sensor node, i.e. GPS, camera, etc.

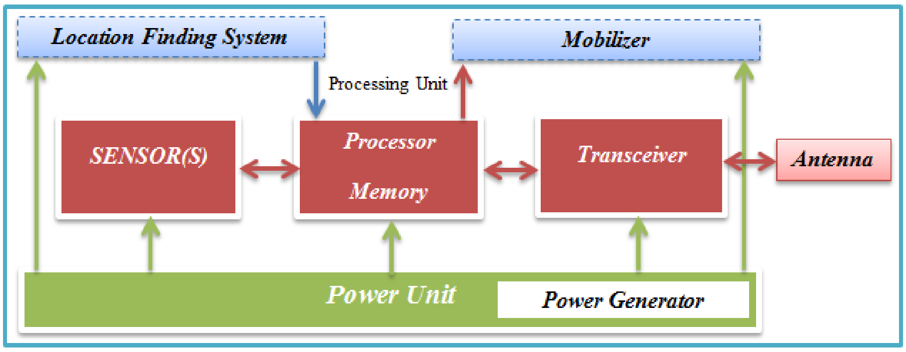

Figure 1.3 General hardware architecture of a sensor node

- ***Fault Tolerance:*** Fault tolerance in WSNs, defines as the ability of the network to sustain its functionalities without any interruption due to sensor node failures (Akyildiz & Canvuran, 2010) or as the level of failures that is allowed by the network to fairly continue its functions (Hoblos, Staroswiecki, & Aitouche, (2000), Shen, Srisathapornphat, & Jaikaeo, (2001)). These failures appear as a result of the hardware constrains or caused by hardware problems. For example, lack of power, physical damage, environmental interference, or software problems.

Fault tolerance can be improved by broadcasting the message to more than a single node in order to utilize in network connectivity in case a failure of the sensor node. Furthermore, redundancy must be used and when designing the protocols and algorithms for WSNs frequent failures of sensor nodes must be addressed. Fault tolerance level differs from application to other means its level depend on the used application.

- ***Scalability:*** Depending on the application, sometimes the number of the sensor nodes that deployed in the WSNs can reach hundreds of nodes. Furthermore, the network should be scalable to accept more nodes, sometimes exceed the thousands of nodes. Therefore, the designed networking protocols must be able to handle these large numbers of nodes efficiently.

- ***Production Costs*:** Sensor node sometimes and according to the used application, needs to be equipped with additional units, like GPS, mobilizer, or power generator.

These units will add additional cost to the sensor node's cost and think the network consist of large number of nodes the result will be unfeasible network because of the high cost of each node. As a result, the network to be practically feasible the sensor cost should be kept low, less than 1$ (Ammer, Rabaey, da Silva, & Patel, 2000).

- ***WSN Topology:*** Topology maintenance is a challenge task especially in case of large number of nodes that must be efficiently deployed in the field that monitor the phenomenon of interest. Topology maintenance is a challenge task in the three phases of the deployment:

1. *Pre-deployment and Deployment Phase*: Sensor nodes can be deployed in two ways, placing one by one or throwing in quantity in the sensor field. As a result, the initial deployment must:

    • Reduce the installation cost
    • Eliminate the need for any pre-organization and preplanning
    • Increase the flexibility of arrangement
    • Promote self-organization and fault tolerance.

2. *Post-deployment Phase:* After the deployment phase, many factors affect and change the topology of the network. Some are for short time, i.e., interference, noise or moving obstacles, some are permanently, i.e., node failures, and others are periodically, i.e., turning a node ON/OFF for some time. All these make the network to work in a different topology than the initial one. As a result, the networking protocols must be able to deal with these changes.

3. Re-deployment Phase of Additional Nodes: According to changes that may happen in post-deployment phase, sometimes nodes need to be added and sometimes nodes need to be redeployed because of the changes in tasks. This is also need special dealing of networking routing protocols.

- ***Transmission Media:*** Reliable communication is a key of the successful operation of WSNs, which communicate wirelessly by creating radio, infrared, or optical links. In order to these networks can interoperable, the chosen medium needs to be known worldwide. Industrial, Scientific and Medical (ISM) bands are the appropriate one due to using free radio, huge spectrum allocation and global availability. Table 1.1 shows some frequency bands that used in ISM applications. Recent Sensor nodes are using 2.5GHz band, which is also supported by IEEE 802.15.4 standard (Akyildiz & Canvuran, 2010).

Table 1.1 Frequency bands available for ISM applications.

Frequency band	Center frequency
6765–6795 kHz	6780 kHz
13 553–13 567 kHz	13 560 kHz
26 957–27 283 kHz	27 120 kHz
40.66–40.70 MHz	40.68 MHz
433.05–434.79 MHz	433.92 MHz
902–928 MHz	915 MHz
2400–2500 MHz	2450 MHz
5725–5875 MHz	5800 MHz
24–24.25 GHz	24.125 GHz
61–61.5 GHz	61.25 GHz
122–123 GHz	122.5 GHz
244–246 GHz	245 GHz

- ***Power Consumption:*** WSN life time mainly depend on the lifetime of limited power source, typically battery operated. Therefore, energy consumption is the main concern in WSNs. Hence, during the operation of each sensor node, the sources that consume energy must be analyzed and maintained efficiently

1.3 Features of Sensor Networks

Wireless Sensor Networks (WSNs) as mentioned above have many benefits over the traditional network. In this section we are going to outline some features of WSNs.

- ***Collaborative Objective*** The objective is the most important aspect of WSN that make it different from other wireless networks (Li, Thai, & Wu, 2008). Normally, its objective is sensing an event in the environment and report the sensed event to a central base station or a sink. The sensor nodes do not compete with each other. However, they collaborate to achieve the certain goal of their deployment. For example, they collaborate to send their data using multi-hop communication in a way that maximize the network lifetime. This is unlike other wireless networks such as wireless local area networks where the nodes (users) are greedy try to maximize their own gains.

- ***Network Scale:*** Although some applications involve a small number of sensors (10-20), other applications may involve a large number of sensor nodes (100-1000) (Akyildiz et al, 2002). Developments in integrated circuit design technology make the mass production of sensor devices relatively inexpensive and this make WSNs with large number of nodes common. Redundancy makes the network more robust to routing and node failures, where, each node has many alternative paths to reach the sink. This is another point that makes WSNs different from other network in terms of scalability.

- ***Many-to-one Communication Paradigm:*** As mentioned above the objective of sensor node is to monitor signal of interest. The events will be reported by the sensor nodes to the base station or the sink where the next action will be decided by. Thus the data flows in upstream (many-to-one); sensor nodes send their reports to the sink, and in downstream (one-to-many); the sink sends queries or control messages to the sensor nodes. This is unlike internet where the traffic flows from a single server to many clients and unlike a peer-to-peer network where the traffic flows between any two nodes of the network.

- ***Nodes with Limited Capabilities:*** The hardware component of sensor node is another difference between WSNs and wireless LAN or any cellular network. Sensor node is not advanced as a wireless laptop, PDA or a cell phone. It is restricted by a battery which is limited in energy and usually cannot be replenished (typically a small lithium battery rated at a few hundred mAh), slower computing speeds (about 4MHz), small memory (about 8KB flash memory and 512 bytes of RAM), low data rates (up to 20 Kbps) and limited communication range (10-100 feet) (Hill et al, 2000). When designing the protocols at different layers, all these limitations that have a direct impact on the functioning of the network must be taken into account.

- ***Clustering for Scalability:*** WSNs consist of large number of nodes. So, distributed protocols for gathering data and arbitrating the access to the wireless channel are needed. These protocols should be scaled well even if the number of nodes has been increased. To achieve this, sensor nodes must be organized in smaller sub-network called clusters which result in lower routing overheads. The clusters could consist of nodes with different hardware capabilities. Within each cluster the responsibilities of coordinating MAC and routing as well as data aggregation could be assigned to nodes with special hardware (Mhatre, Rosenberg, Kofman, Mazumdar, & Shroff , 2005).

- ***Node Deployment versus Placement:*** Depending on the application, Sensor nodes can be either thrown randomly en masse over the area of interest (battlefield surveillance, forest fire detection, etc.), or placed one by one at specified locations (temperature and light monitoring in buildings, seismic monitoring of bridges and buildings, etc.). In this case ensuring network connectivity is relatively easy. However, in the first case (randomly deployment) to ensure network connectivity, a certain extent of over-provisioning of nodes is required**.**

- **Node Mobility and Dynamic Topology:** Although some in applications sensor nodes are static, in many applications, such as monitoring of military personnel and equipment and animals monitoring, the nodes are mobile. Hence, according to these mobile nodes, the topology of network will change and the routing information has to be updated which result in a dynamic network topology.

In some application in order to save power the nodes need to turn off its transceiver and enter a sleep state, and accordingly the topology of the network will be changed and also due to node failures. Hence, sensor networks often have a dynamic topology because of node mobility, node failures, and radio duty-cycling. Not to forget, when designing the communication protocols, the highly mobile nodes have a stronger impact on the network topology than the other factors.

CHAPTER TWO
ARCHITECTURE

2.1 Single-Node Architecture

The basic unit of the wireless sensor network is a sensor node. Its duty not only senses the environment and the physical world, furthermore it contains other units helping in processing, and delivering the sensed data (Wang, 2010).

Sensor node, in addition to sensor unit, communication unit, controller unit, and memory and power unit, depending on application scenarios and requirements maybe include other units like GPS, camera, energy scavenge and locomotive units. See figure 2.1 for basic units of sensor node.

2.1.1 Hardware Components

Depending on the application's requirements, the hardware components of a wireless sensor node must be chosen regarding to size, cost and energy consumption (Karl & Willig, 2005). For example in some application the size, weigh, price, and energy consume must not exceed 1 cc, 100g, US$1, and 100 µW consecutively (Rabaey, Ammer, da Silva, Patel, & Roundy, 2000). In some application the size is important whereas in other application the power supply and cost are more important. As a result, there is no such standard available to support all applications.

A basic sensor node consists of the following five components (Figure 2.1):

***Controller*:** A controller for data processing and code executing.

***Memory*:** Some memory for programs and intermediate data storing.

***Sensors and actuators*:** For observing and controlling the physical parameters of the environment.

***Communication*:** A transceiver for sending and receiving information over a wireless channel.

Power supply: To provide the necessary energy to other components to fulfill their tasks.

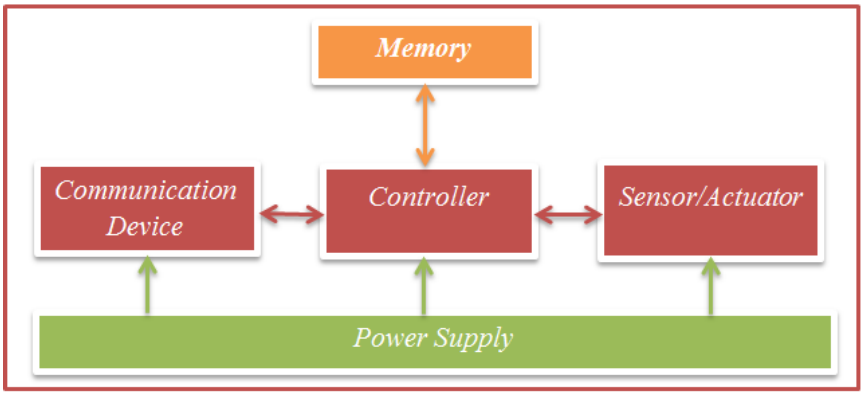

Figure 2.1 Overview of main sensor node hardware components

Each of these components must use the energy in an effective way while they doing their tasks (Karl & Willig, 2005). For instant, some of these components like communication device and the controller must go to sleep state most of the time specially when there is no event to process. However, they should be able to wake up again. Accordingly, a sensor should be able to detect the events that exceeds the threshold values and makes an interrupt. As a result, achieving these functions, interconnection between these individual components is required.

- ***Controller (Microcontrollers):*** The microcontroller, just like the Central Processor Unit (CPU) of a desktop computer, is the core of a wireless sensor node but it consumes less energy than CPU. Its functions are collecting data from the sensors, processing this collected data, deciding when and where to send it, receiving data from other sensor nodes and deciding on the actuator's behavior (Wang, 2010). Microcontrollers are flexible in connecting with sensors, often have built in memory, programmable and finally they able to going into sleep mode. Hence, reduce their power consumption (Karl & Willig, 2005).

- ***Memory:*** RAM is fast. However, opposite ROM, loses its content if power supply is interrupted. Therefore, in WSN the memory component includes both the on-chip random access memory (RAM) used by the microcontroller to store intermediate sensor readings and packets from other nodes and the on-board read-only memory (ROM) used for storing program codes. Here, ROM typically includes Electrically

Erasable Programmable ROM (EEPROM) and flash memory. Flash memory sometimes serve as intermediate storage of data in case RAM is insufficient or when the power supply of RAM should be shut down for some time. Depending on the application requirements, size of memory is ranging from hundreds of KB to hundreds of MB (Wang, 2010).

- ***Communication Device:*** The individual nodes exchange data between each other via communication device. Wireless communication is more preferable than wired communication because the last limits the flexibility and scalability of a sensor network (Wang, 2010). Long range and high data rates and acceptable error rates at reasonable energy expenditure provided by Radio Frequency (RF) has made it the preferred one in communication that meet all the requirements of WSNs applications. Moreover, it does not require line of sight between sender and receiver.

Transceiver is such a device that combines both a transmitter and a receiver. It converts a bit stream coming from a microcontroller to and from radio waves. A range of low-cost transceivers is commercially available that incorporate all the circuitry required for transmitting and receiving – modulation, demodulation, amplifiers, filters, mixers, and so on. According to communication needs, it operates in four operational states (Raghunathan et al, 2002):

  - *Transmit***:** the transmit part of the transceiver is active.
  - *Receive***:** the receive part is active.
  - *Idle:* some parts of the circuitry are active and others are switched off, means the transceiver is ready to receive.
  - *Sleep*: significant parts of the transceiver are switched off.

- ***Sensors and Actuators****:* Actual sensors and actuators are constructing the wireless sensor network entirely.

- *Sensors:* is a device that represents an interface between physical and electrical word. It is a device which senses the physical environment (such as temperature, pressure, light, sound, etc.) and convert this sensed physical signals to an electrical signals that can be treated by digital environment, such as computers to make people easy understand, monitor and control machines and environments

(Wang, 2010). Sensors represent very important part of our life you can see them every day in everywhere in lamp and sensitive buttons. It used in applications include cars, medicine, machines, airplanes and application most people never aware. Sensors can be roughly categorized into three categories (Raghunathan et al, 2002):

1. *Passive, omnidirectional sensors.* These sensors can measure a physical quantity without effecting the environment, they are passive. There is no notion of “direction” involved in these measurements. Typical examples for such sensors include thermometer, light sensors, vibration, microphones, humidity, mechanical stress or tension in materials, chemical sensors, smoke detectors and air pressure.

2. *Passive, narrow-beam sensors.* These sensors are passive as well, but have a well-defined notion of direction of measurement. A typical example is a camera, which can “take measurements” in a given direction and can rotated if needed.

3. *Active sensors.* These sensors actively probe the environment. For example, a sonar or radar sensor or some types of seismic sensors, which is generates shock waves by small explosions. In practice, sensors from all of these types are available in many different forms with many individual peculiarities including accuracy, dependability, energy consumption, cost, size, and so on.

- *Actuators:* opposite the sensors, convert electrical signals into some action. They are mechanical devices for moving or controlling a mechanism or system. Moreover, they are devices that accept electrical signal and make changes in physical domain by generating motion, force, etc. Actuators, for the purposes of designing a WSN, are as diverse as sensors. They are a bit simpler to take account of. In principle, they use to open or close a switch or a relay or to set a value in some way in order to control a motor, a light bulb, or some other physical object that is not really of concern of the way that communication protocols are designed.

- ***Power supply of sensor nodes:*** Power supply is a crucial system component of the untethered wireless sensor nodes. Storing power is conventionally done by using batteries, which is the power source of sensors. For example, a normal AA battery stores about 2.2–2.5 Ah at 1.5 V. Battery design is a science and industry in itself and energy scavenging has attracted a lot of attention in researches.

2.1.2 Energy Consumption of Sensor Nodes

Recently wireless sensor networks have emerged as an effective way of monitoring physical environments. The main challenges in these networks are the constrained energy and computational resources of the sensor nodes, and these constrains have to be taken into account at all levels of system hierarchy.
As seen, one of the most important requirements is that, Wireless sensor architectures and applications must be provided or developed with low energy consumption protocols that make well-use of the limited energy of the sensor nodes required (Slijepcevic & Potkonjak, 2001). Furthermore, sensor nodes must avoid direct communication with a distant destination and it is better to send the messages in multi-hop than sending it in a single hop (Bouabdallah, Bouabdallah, & Boutaba, 2009).

The controller, memory and the sensors are the main consumers of energy. To reduce power consumption of these components it is good to start with designing low-power chips for an energy-efficient sensor node. However, this is not enough, the components must operate properly, where the wireless sensor node most of the time has nothing to do and it is best to turn it off. Completely turning off a node is not possible because it should be able to wake up again. Some modes can be introduced for all components of a sensor node, for a controller, typical states are “active”, “idle”, and “sleep”; a radio modem could turn transmitter, receiver, or both on or off. Sensors and memory could also be turned on or off.

- ***Microcontroller Energy Consumption:*** Embedded controllers, commonly implement the concept of multiple operational states as outlined above, it is also fairly easy to control. To understand the idea takes this example:

Intel StrongARM

The Intel StrongARM (Intel product documentation, 2000) provides three sleep modes:

- In **normal mode**, all parts of the processor are fully powered. Power consumption is up to 400 mW.
- In **idle mode**, clocks to the CPU are stopped; clocks that pertain to peripherals are active. Any interrupt will cause return to normal mode. Power consumption is up to 100 mW.
- In **sleep mode**, only the real-time clock remains active. Wakeup occurs after a timer interrupt and takes up to 160 ms. Power consumption is up to 50 μW.

- ***Memory:*** The power needed to drive on-chip memory is usually included in the power consumption numbers given for the controllers. Therefore, the most relevant kinds of memory are on-chip memory of a microcontroller. FLASH memory off-chip RAM it is rarely used because it influences node lifetime. For example, consider the energy consumption necessary for reading and writing to the Flash memory used on the Mica nodes (Mainwaring, Polastre, Szewczyk, Culler, & Anderson, 2002). Reading data takes 1.111 nAh, while writing requires 83.333 nAh. As shown, writing to FLASH memory can be a time- and energy-consuming task that is best avoided if it possible. For detailed numbers, it is necessary to consult the documentation of the particular wireless sensor node and its FLASH memory under consideration.

- ***Radio Transceivers:*** Transmitting and receiving data between a pair of nodes are two tasks of a radio transceiver. It, like microcontrollers, can operate in different modes. The simplest ones are being turned on or turned off. To reduce energy consumption, the transceivers should be turned off most of the time and only be activated when necessary.

- ***Power Consumption of Sensor and Actuators:*** Because of the wide diversity of the actual sensors and actuators it is impossible to provide any guidelines about the power consumption. However, as an example, passive light or temperature sensors – the power consumption can perhaps be ignored in comparison to other devices on a wireless node. In contract, active devices like sonar, power consumption can be quite considerable and must be considered in the dimensioning of power sources on the sensor node. It requires a look at the intended application scenarios and the intended sensors to be used in order to derive any meaningful numbers.

2.2 Network Architecture

2.2.1 Sensor Network Scenarios

Wireless sensor network consists of Wireless sensor nodes, which monitor the environment and produce data, and sink/sinks, which collects/collect data from the sensor nodes and does not produce any data. Sink sometimes works as a gateway to another network like internet (Wang, 2010).
Depending on the capabilities of the sensor nodes and sinks and communication paradigm used by the sensor nodes and sink, wireless sensor networks can work in different architectural and operational scenarios. For instant, sometimes wireless sensor node has more advanced units that enable it to take more responsibilities inside the WSN. Bellow, several typical sensor network scenarios are introduced.

2.2.1.1 Types of Sources and Sinks

A source is any entity in the network that can provide information, here, typically a sensor node or an actuator node that provides feedback about an operation.
A sink is the entity where information is required. There are three options for a sink:

- A sink could be just another sensor/actuator node that belongs to the sensor network.
- A sink could be an actual device outside this network, for example, it could be a PDA used to interact with sensor network.
- A sink could be a gateway to another larger network such as the Internet.

Figure 2.2 shows the sources and the main types of sinks.

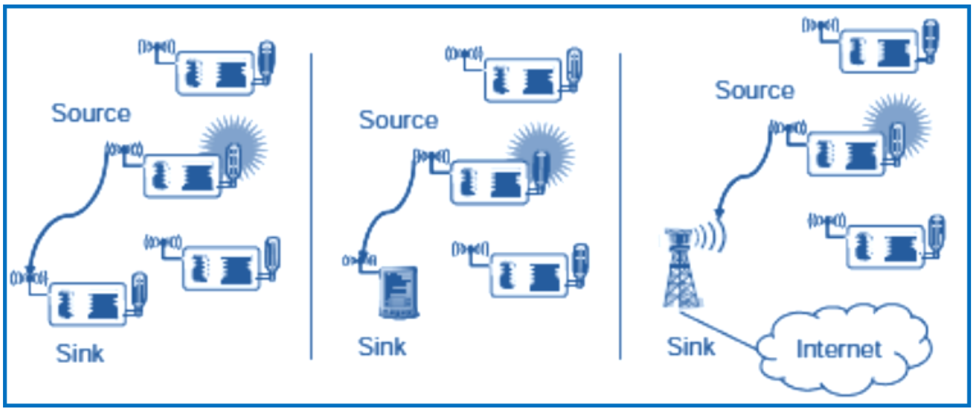

Figure 2.2 Three types of sinks in a very simple, single-hop sensor network

2.2.1.2 Single-Sink Single-Hop WSN

Scalability, defined in part 2.2.2.4, is the critical problem of this scenario, where by increasing the number of nodes the amount of data gathered by the sink increases and once its capacity is reached the network size cannot be increased any more.

To calculate the maximum number of nodes that a sink can serves:
Let ***N*** is number of nodes, ***R*** the channel bit rate, ***α*** is factor overhead introduced by all protocol stack layers (takes value between 0.5 and 0.1), nodes are requested to send their samples (each sample = ***D*** bytes) taken from the monitored space every ***T*** seconds.

Under such assumptions, the application throughput will be approximately equal to ND8/TR. Then, we reach the following inequality: $ND8/T \leq R\alpha$; and then

$$\mathbf{N \leq R\alpha T/(8D)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\mathbf{(2.1)}$$

For example,
Assume R = 250 Kbit/s, T = 1 s, $\alpha = 0.1$ and D = 3 then,
$N \leq 250000 \times 0.1 \times 1 * (8 \times 3)$ = 1041 approximately

If T is 0.01 then the maximum number of nodes will not exceed 10 (Verdone, Dardari, Mazzini, & Conti, 2007).

Figure 2.3 shows the traditional single-sink WSN.

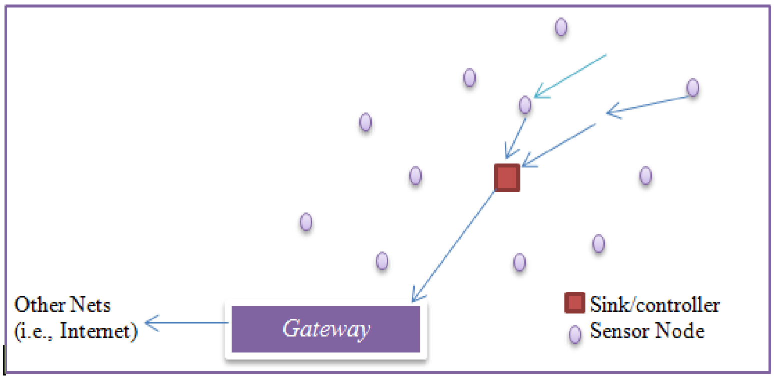

Figure 2.3 Traditional single-sink WSN

2.2.1.3 Single-Sink Multi-Hop WSN

In this scenario a node can reach the sink through multiple hops. Let the average number of hops that a node can send a data sample = H then the total number of sensors in a single-sink multi-hop WSN:

$$\mathbf{N \leq R\alpha T/(8DH)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \mathbf{(2.2)}$$

This means that the capacity of the network will be reduced by a factor of H

2.2.1.4 Multi-Sink Multi-Hop WSN

Multiple-sink WSN opposite single-sink WSN can be scalable, means the same performance can be achieved even by increasing the number of nodes. In this scenario (Figure 2.4) the probability of isolated clusters of nodes that cannot deliver their data owing to unfortunate signal propagation conditions will be decreased. This ensures better performance of network. However, communication protocols are more complex and should be designed according to suitable criteria. If we assume S is the total number of sinks in the network and by expressions (2.1) and (2.2). Each sink can serve up to N nodes:

$$\mathbf{N \leq SR\alpha T/(8DH)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \mathbf{(2.3)}$$

Taking the same example above, where R = 250Kbit/s, T = 10msec, $\alpha = 0.1$ D = 3 and S = 5

$N \leq 5 \times 250000 \times 0.1 \times 1/(8 \times 3 \times 2) = 26$ approximately

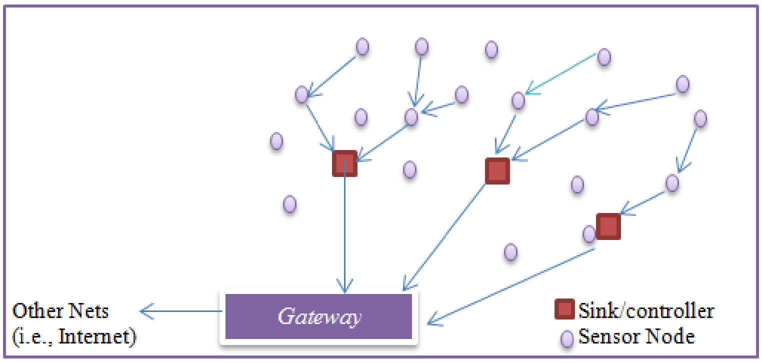

Figure 2.4 Multi-sink WSN

2.2.1.5 Single-Hop versus Multi-Hop Networks

Because of power limitation of radio communication and a limitation on the feasible distance between a sender and a receiver, in WSN, direct communication between source and sink is not always possible. Because of the huge number of nodes that cover the ground, for instant, in environmental or agriculture applications. This obstacle could be overcome by using relay stations, in which the packets take multi hops from the source to the sink. Moreover, to achieve energy efficiency, sensor nodes communicate in multi-hop network to forward messages to the sink because achieving a reliable transmission with a distant destination needs high transmission power (Akyildiz et al, (2002), Kredo II & Mohapatra, (2007).

This concept is illustrated in (figure 2.5) and it is attractive for WSN. Since the sensor nodes themselves can act as such relay without the need to additional device.

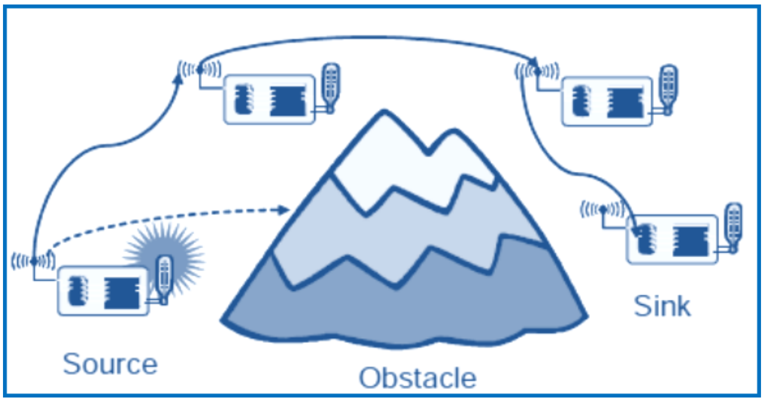

Figure 2.5 Multi-Hop Networks with Obstacle

2.2.1.6 Three Types of Mobility

One of the main benefits of wireless communication is its ability to support mobile participants. In wireless sensor networks, mobility can appear in three forms:

- *Node mobility:* Depend on the application of WSN the nodes themselves could be mobile. In this situation, the network has to reorganize itself frequently enough to be able to function correctly. This shows that there are trade-offs between the frequency and speed of node movement on the one hand and the energy required to maintain a desired level of functionality in the network on the other hand. An example for this kind of node mobility is in livestock surveillance, where sensor nodes attached to cattle,
- *Sink mobility:* When a mobile requester requests a data that is not locally available but it must be retrieved from a remote part of the network. And since the requester can communicate only with neighbor nodes, it has to move to that remote part of the network. Here the network, possibly with the assistance of the mobile requester, must make provisions that the requested data actually follows and reaches the requester despite its movements (Shen, Srisathapornphat, & Jaikaeo, 2001).

Information requesting by a human user, for instant by a PDA (mobile sink), where he is not part of sensor network, while walking in an intelligent building is a good example for this kind of mobile information sink. Figure 2.6 illustrate the mobile sink.

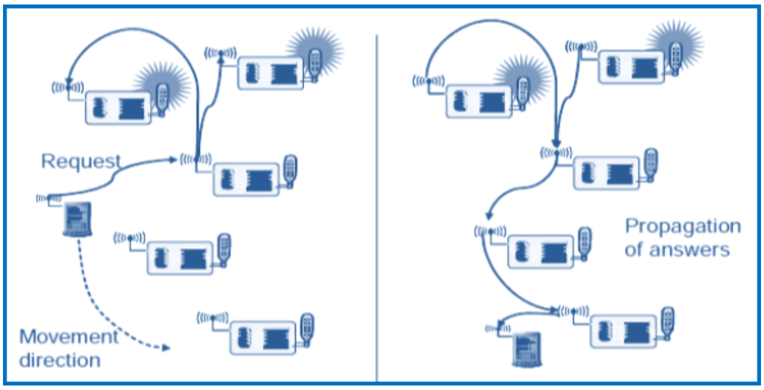

Figure 2.6 A mobile sink moves through a sensor network

- **Event mobility:** An example for this kind of mobility is in applications like event detection, the cause of the events or the objects to be tracked can be mobile. Usually the observed events covered by number of sensors at all time. So, sensors will wake up around the object to observe it and then go back to sleep. As the event source moves through the network, it is accompanied by an area of activity within the Network. This notion is described by Figure 2.7, where the task is to detect a moving elephant and to observe it as it moves around (dashed line indicate the elephant's trajectory; shaded ellipse the activity area following or even preceding the elephant).

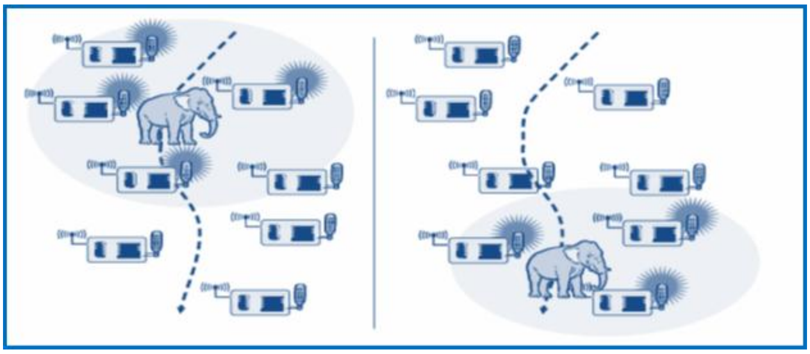

Figure 2.7 Area of sensor nodes detecting an event – an elephant movement (Intanagonwiwat, Govindan, Estrin, Heidemann, & Silva, 2003)

2.2.2 Optimization goals and figures of merit

Although different types of applications and different forms of network solution are found, questions like how to optimize a network, how to compare these solutions, how to decide which approach better supports a given application, and how to turn relatively imprecise optimization goals into measurable figures of merit, are impossible to answer because of the huge number of applications. Below are some aspects:

2.2.2.1 Quality of Service

High-level QoS attributes in WSN, just like in traditional networks, highly depend on the application. Some generic possibilities are (Karl & Willig, 2005):

- *Event detection/reporting probability:* What is the probability that an event that actually occurred is not detected or, more precisely, not reported to an information sink that is interested in such event? Simply, this probability can depend on reporting of such event (e.g. routing tables) or depend on the run-time overhead (e.g. sampling frequencies).

- *Event classification error:* If events are not only need to be detected but also need to be classified, the error in classification must be small.

- *Event detection delay*: What is the delay between detecting an event and reporting it to any or all interested sinks?

- *Missing reports:* The probability of undelivered reports should be small in applications that require periodic reporting.

- *Approximation accuracy:* For function approximation applications, what is the average/maximum absolute or relative error with respect to the actual function?

- *Tracking accuracy:* Tracking applications must not miss an object to be tracked, the reported position should be as close to the real position as possible, and the error should be small.

2.2.2.2 *Energy Efficiency*

The most important issue in WSN is the energy, the limited energy capacity of sensor nodes, usually battery operated, dictates how communications must be performed inside wireless sensor networks (WSNs). To achieve energy efficiency, sensor nodes communicate in multi-hop network to forward messages to the sink because achieving a reliable transmission with a distant destination needs high transmission power. Moreover, a sensor node to reduce the communication burden may process and aggregate incoming data before relaying it to its neighbor node (Sankarasubramaniam, Akyildiz, & Mchughlin, 2003). Two major aspects have to be examined in order to determine the optimal data packet size for communication between neighboring sensor nodes:

1) Using Energy efficiency as optimization metric.
2) Effect of retransmissions, error control parities and encoding/decoding energies on energy efficiency.

Furthermore, the energy efficiency depends on both channel conditions and energy consumption characteristics of a sensor node.

2.2.2.2.1 Energy Consumption Characteristics. In WSN the smallest communication entity between adjacent sensor nodes is the link layer data packet. As shown in figure (2.8) link later data packet is consists of header field (long **α** bit), payload (size **ɭ** bit) and trailer (**τ** bit long).

Header (α)	Payload (ɭ)	Trailer (τ)

Figure 2.8 Link Layer Packet Format

Header identifies event, location or attribute so α is just few bytes. The payload contains information bits and the trailer is composed of parity bits for error control (Sankarasubramaniam et al, 2003).

Based on energy model in (Shih et al, 2001), the energy required to communicate one bit of information (E_b) through a single hop is:

$$E_b = E_t + E_r + \frac{E_{dec}}{l} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.4)$$

Where:

E_{dec}: It is the decoding energy per packet

E_t : It is the transmitter energy consumption

E_r : It is the receiver energy consumption. E_t And E_r are given by

$$E_t = \frac{\left((P_{te} + P_0)\frac{(l+\alpha+\tau)}{R} + P_{tst}T_{tst} \right)}{l} \ldots\ldots\ldots\ldots\ldots\ldots (2.5)$$

$$E_r = \frac{\left((P_{re})\frac{(l+\alpha+\tau)}{R} + P_{rst}T_{rst} \right)}{l} \ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.6)$$

Where:

P_{te} / P_{re} : Power consumed in the transmitter/receiver electronics

P_{tst} / P_{rst} : Start-up power consumed in the transmitter/receiver

T_{tst} / T_{rst} : Transmitter/receiver start-up time

P_0 : Output transmit power

R : Data arte (20 Kbps)

Energy, as discussed, is a precious resource in wireless sensor networks and therefore, the energy efficiency should make an evident optimization goal and should be carefully distinguished to form actual and measurable figures of merit. The most commonly considered aspects are:

- *Energy per correctly received bit:* Amount of energy spent on average to transport one bit of information from the source to the destination, is a useful metric for periodic monitoring applications.

- *Energy per reported (unique) event:* Similarly, what is the average energy spent to report one event? Since the same event is sometimes reported from various sources, it is usual to normalize this metric to only the unique events

- *Delay/energy trade-offs:* Applications that have a notion of "urgent" events can increase energy investment for a speedy reporting of such events. Here, the trade-off between delay and energy overhead is interesting.

2.2.2.3 Network Lifetime

Network lifetime is a critical concern in the design of WSNs. In many applications, replacing or recharging sensors sometimes is impossible. Therefore, many protocols have been proposed to increase network lifetime. It is difficult to analysis network life time because it depends on many factors like network architecture and protocols, data collection initiation, lifetime definition, channel characteristics, and energy consumption model. Below are the most important network characteristics that affect the network lifetime.

- *Network Architecture***.** Specifies how sensors should report their data to the Access points. For example in flat ad hoc architecture, it is done by multiple hops. In hierarchical WSNs, it is done by cluster heads, where the sensors form clusters and report their data to the cluster heads which in turn send it to Access points and so on.

- *Data Collection Initiation.* Data collections in a WSN can be initiated according to the applications by the event of interest (internal clock of sensor) or by demanding from the end user (request from Access point).

- *Channel and Energy Consumption Model***.** Energy consumption in WSN can classify into two main categories: first, continuous energy consumption and second reporting energy consumption. The first is the minimum energy that sustain network during its lifetime, and the second is the energy that consumed during data collections, transmission, reception, and possibly channel acquisition.

- *Lifetime Definition***.** Network lifetime is the time span from the deployment to the situation that the network is nonfunctional. And here some other definition of the lifetime:

    - *Time to first node death***:** When does the first node in the network run out of energy or fail and stop operating?
    - *Network half-life***:** When have 50% of the nodes run out of energy and stopped operating?
    - *Time to partition:* When does the first partition of the network in two (or more) disconnected parts occur? This can be as early as the death of the first node or occur very late if the network topology is robust.
    - *Time to loss of coverage:* The first time any spot in the deployment region is no longer covered by any node's observations. In tracking applications, for example, *r* redundant observations are necessary, the corresponding definition of loss of coverage would be the first time any spot in the deployment region is no longer covered by at least *r* different sensor nodes.
    - *Time to failure of first event notification:* A network partition can be seen as irrelevant if the unreachable part of the network does not want to report any events in the first place. This can be due to an event not being noticed because the responsible sensor is dead or because a partition between source and sink has occurred.

Obviously, the longer these times are the better does a network performs. However, general formula for network lifetime has been driven in (Chen & Zhao, 2005) which hold independently of the characteristics that affect the network lifetime mentioned above (network architecture and protocols, data collection initiation, lifetime definitions, channel characteristics, and energy consumption model). This general formula depends on two physical parameters: the channel state and residual energy of sensors. It indicates that channel state information (CSI) and the residual energy information (REI) should be exploited in the lifetime maximizing protocols.

By using both CSI and REI, the proposed protocol maximizes the minimum residual energy across the network in each data collection. The average lifetime of WSN have studied in a general setting, no network architecture has specified nor the channel and the energy consumption model. Moreover, the interesting thing is the obtained formula applies to any definition of the network lifetime. The theorem of the general formal that mentioned above, which is driven in (Chen & Zhao, 2005), is as below:

For a WSN with total non-rechargeable initial energy E0, the average network lifetime E[L], measured as the average amount of time until the network dies, is given by

$$E[L] = \frac{E_0 - E[E_w]}{P_C + \lambda E[E_r]}$$

Where Pc is the constant continuous power consumption over the whole network, E[Ew] is the expected wasted energy (i.e., the total unused energy in the network when it dies), λ is the average sensor reporting rate defined as the number of data collections per unit time, and E[Er] is the expected reporting energy consumed by all sensors in a randomly chosen data collection(Chen & Zhao, 2005, p. 977)

Finally, lifetime maximizing protocol (max-min protocol) aims to reduce the average wasted energy E[*Ew*] by exploiting the REI of individual sensors and the average reporting energy E[*Er*] by exploiting the CSI to give the priority to the sensors with better channels transmission. Hence, energy consumed in transmission will be reduced.

2.2.2.4 Scalability

Scalability is the ability to maintain performance characteristics irrespective of the size of the network. Because of the huge number of nodes in WSN, scalability is an indispensable requirement. The need for extreme scalability has direct consequences for the protocol design. Architectures and protocols should implement appropriate scalability support rather than trying to be as scalable as possible.

2.2.2.5 Robustness

Wireless sensor networks should not fail just because a limited number of nodes run out of energy, or because their environment changes and severs existing radio links between two nodes. They should exhibit an appropriate robustness and these failures must be solved by finding other route. A precise evaluation of robustness is difficult in practice and depends mostly on failure models for both nodes and communication links.

CHAPTER THREE
WSNs APPLICATIONS

WSNs, as mentioned before, consist of sensor nodes and sinks, a sensor node which in turn consists of sensor unit, communication unit, controller unit, memory and power unit and may equipped with various type of sensors like seismic, magnetic, thermal, visual, infrared, acoustic, and radar, that are able to monitor different types of physical phenomenon like temperature, humidity, pressure, movement, light, soil makeup, noise levels, and mechanical stress levels on attached objects. As a result WSNs have a wide range of applications, they are as various as sensors. These applications mainly involve monitoring and controlling the environment. The concept of WSNs is simply based on this equation (Rudas, Fodor, & Kacprzyk, 2009).

Sensing + CPU + Radio = Thousands of potential applications

Hundreds of application will spring to mind once the people understand the capability of WSNs. However, mainly, WSNs are categorized into five categories (Akyildiz & Canvuran, 2010).

1. Military application
2. Environmental applications
3. Health applications
4. Home and building application (automations)
5. Industry applications.

3.1 Classification of WSNs Application

According to WSNs application objectives, traffic characteristics and data delivery requirements, WSNs can be classified into, and current WSN applications fall under one of these, following four classes (Li et al, 2008).

3.1.2 Event Detection and Reporting

Common characteristic of the applications that belong to this class, is the infrequency of occurrence the events of interest. Military surveillance, fire detection, and detecting odd behavior or failures in a manufacturing process are some applications example of this class.

Sensor nodes in these applications are expected to be inactive most of the time, they triggered when an event is detected. The reported event by the node/nodes to the sink/sinks contains some location information about the event, and a description of the event nature. Such networks from the point view of the application level have important problem that is to minimize the probability of false alarms. However, from the point view of networking the problem is during routing the event report, the time (the process) of event detection may be over.

3.1.3 Data Gathering and Periodic Reporting

In this class, each sensor constantly produces some amount of data that has to be relayed to the sink/sink. Monitoring the environmental conditions that affect crops or livestock, monitoring temperature, humidity and lighting in office buildings, are application examples for this class. These WSNs applications maybe include actuators as a control, for instance, to ON or OFF a switch. The reported event might be contains some location information if the sink is interested in recreating a spatial profile of the readings.

3.1.4 Sink-initiated Querying

In some monitoring applications like applications that mentioned in the previous subsection (Monitoring environmental conditions that affect crops or livestock, monitoring temperature, humidity and lighting in office buildings), the sink may query a set of sensors for their measurements rather than each sensor periodically reporting it. Hence, the sink is able to extract information at a different resolution or granularity, from different regions in space.

In monitoring application, a manufacturing process for example, the sensors could report an event whenever there is unexpected behavior. Then the sink can ask some specific set of sensors to obtain more information and according to this information the sink may leads the appropriate actuators or give an alarm for human intervention. Here, the important issue is that communication protocols need an effective means to address and route data to and from dynamic sets of sensors.

3.1.5 Tracking-based Applications

This class is combines some of the characteristics of the above three classes. For example when an event is detected, the sensor node reports it to the sink. Then, the sink may initiate queries to receive time-stamped location estimates of the target in order to calculate the trajectory and keep querying the appropriate sets of sensors. Military or border surveillance applications to track an intruder or any undesirable movements and environmental applications like tracking the movements and patterns of birds or small animals are some examples for this class.

Communication protocols must design according to the answer of these questions: is it better to query, compute and route on the fly or is it better to maintain some level of organization or connectivity to streamline the process of tracking.

3.2 Application Areas And Scenarios

This section gives an overview of the main applications of WSNs .The application areas that taken into consideration are the following (Verdone et al, 2007):

- Environmental applications
- Military application
- Health applications
- Home applications (automations)
- Industry applications
- Other commercial applications

For each application area, there are some scenarios which describe a situation of using different sensors to monitor an environment, provide health services, improve an industrial activity, etc. Below, the application areas and their scenarios are introduced.

3.2.2 Environmental Monitoring

Environmental monitoring applications are important for society, since these applications can monitor indoor or outdoor environments. Supervising thousands square kilometers of area need duration of time maybe took years. However, using WSN it is possible to obtain localized measurements and detailed information about natural spaces that it is not possible to do this through known methods. WSNs provide security and surveillance concerns for natural disasters such as floods and earthquakes. By installing it closer to places where these phenomena may occur and because these applications require real-time monitoring technologies with high security requirements, the network should respond to the changes of the environment as quick as possible.

Inability of humans to be present all the time in the supervised areas was one of the first ideas behind using WSNs in environment monitoring. An environmental monitoring application may be used in either a small or a wide area for the same purposes. Such networks have to be infrastructure-less and very robust, power efficient, fault tolerant, and scalable in the order of tens or hundreds nodes because of the inevitable challenges in nature, such as living things or atmospheric events.
The following scenarios are related to environmental applications:

- Forest fire detection

- Monitoring volcanic eruptions
- Animal tracking: **ZebraNet**
- Agricultural application

- ***Forest fire detection:*** One of the main problems of forest fires it is, when the fire becomes large it becomes very difficult to put out the fire and sometimes impossible. So, WSN could be deployed to detect a forest fire in its early levels where each node can gather different information temperature, humidity, pressure and position and send it by multi-hop communication to the control center through a number of gateway devices that connected to mobile networks (e.g., Universal Mobile Telecommunications System – UMTS) and distributed throughout the forest (Verdone et al, 2007). Once a fire-related event is detected, such as sudden temperature rise, the control center will be alarmed and the person who in charge will check whether it is false alarm by using the data collected from other nodes or sending a team to check the situation locally. If there is any fire both fire-fighters and helicopters will be sent to extinguish the fire before it grows and involves all the forest, see figure 3.1.

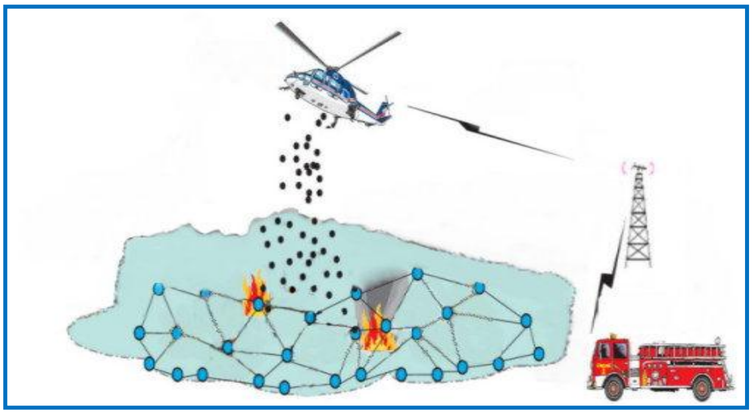

Figure 3.1 WSNs forest fire detection

- **Volcano Monitoring:** Volcano Monitoring is one of the applications that continuous human access is impossible. WSN can be easily deployed near active volcanoes to continuously monitor their activities and provide data at a scale and resolution (Akyildiz & Canvuran, 2010). As a proof of concept of WSN applications in volcano monitoring, two case studies were conducted on two volcanoes in Ecuador during 2004–2005 (Werner-Allen et al. 2006). The used sensor nodes were equipped with higher gain external antennas to improve the communication range and three long-haul communication nodes were used to transmit the data to a central controller covering a 3 km array. To collect information and manage the network remotely, a laptop equipped with a directional antenna was used as a sink. The application aimed to collect seismic information through occurred earthquakes, normally last less than 60 seconds, near the volcanoes (figure 3.2). Therefore, the used seismic sensors were high sampling rate (100Hz). When the nodes detect an event they report it to the sink.

To provide location information, a GPS unit equipped with a MicaZ node. So, important information related to the physical processes at work within a volcano's interior had been provided.

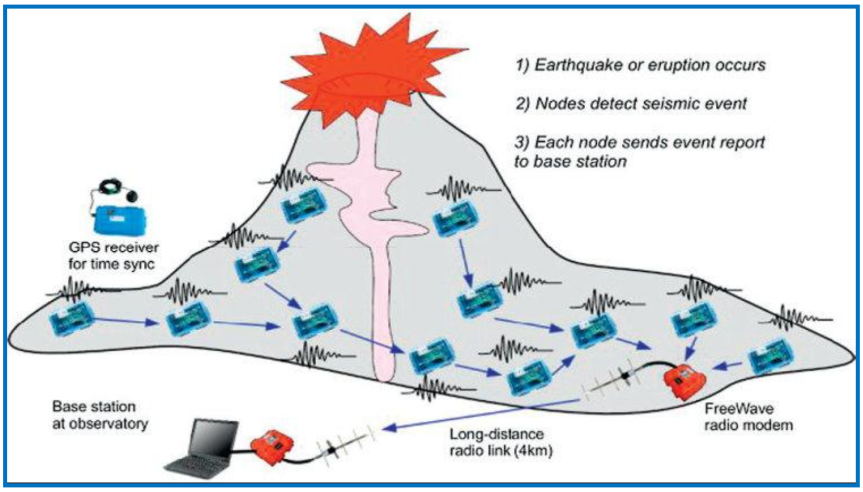

Figure 3.2 Volcano monitoring

- **ZebraNet:** Is an animal tracking system deployed in Kenya to investigate the long-term movement patterns of zebras, their interactions within and between species, as well as the impacts of human development (Juang et al, (2002), Zhang, Sadler, Lyon, & Martonosi, (2004)) and can be characterized as a highly mobile sensor network without a static sink. This system consist of collars, contain a GPS unit, associated microcontroller, two types of radios (short range and long range), a lithium-ion polymer high-energy-density battery, and a solar array for recharging the battery, attached to the zebras to collect information about their location (figure 3.3 taken from (Banerjee & Khuller, 2001)). These collars through GPS unit send every 3 minutes information about their location, this information collected by base station that is hold by the researchers during their trips inside the field.

Figure 3.3 The ZebraNet sensor collar attached to a zebra

The location information is transmitted whenever there is opportunity communicate with the base station, i.e., it is transmitted in a *delay-tolerant* manner. For example, sometimes, a node could be out of communication range of the mobile sink for long time. So a data sharing policy is adopted in ZebraNet where each sensor node shares the collected information with its neighbors. The location information of a node as well as its previous neighbors is downloaded the moment it establishes a connection with the sink. The location information of the zebras collected by a group of sensors provides the domain scientists with worthy information to model the movement of these animals. Furthermore, it contributes to the domain sciences, and used to develop protocols according to the mobility patterns of the zebras.

- **Agricultural Application:** WSN used in the fields to find fertile, soil moisture, sun heat, depth of water, soil water tension, and air passing percentage in plants, in addition to other properties for plant to grow normally, healthy and fast. See figure 3.4, where, in order to monitor the temperature, light and soil moisture, a large number of nodes are deployed. The nodes periodically collect data and report it to the sink that located in the house (Nayak & Stojmenovic, 2010). After getting this information we can calculate the rate of raising planet or trees and enhance the agricultural production. Also it used for management applications, examples of these applications the COMMON-Sense Net (CSN) proposed in (Depienne, 2007) its goal is to enable an appropriate water management system.

Figure 3.4 application of WSN in agriculture

3.2.3 Military applications

Characteristics of WSNs, like rapid deployment, self-organization, and fault tolerance, make them a very promising sensing technique for military systems (Akyildiz & Canvuran, 2010). It can be a complementary part of military C*ommand*, *Control*, *Communications*, *Computing*, *Intelligence*, *Surveillance*, *Reconnaissance*, *and Targeting* (C4ISRT) system. Moreover, this system (C4ISRT) is capable of automatic self-organization and calibration, and use filter to avoid both environmental and sensing noises of inexpensive sensors. In battlefield, WSNs devices able to sense acoustic, magnetic signals generated by different target objects, perform detection and track the targets as well as sending the real time enemy mobility information to a command center. Sensors in an intelligent way can be put inside the vehicles, tanks, fighter planes, submarines, missiles, or torpedoes to guide them around obstacles to their targets. WSNs in military can be used in such applications (Karlof & Wanger, (2003), Boomerang shooter detection system, Lédeczi et al, (2005), and He et al, (2006) :

- Sniper detection System
- Monitoring friendly forces, equipment, and ammunition.
- Battlefield surveillance.
- Reconnaissance of opposing forces and terrain.
- Battle damage assessment.
- Nuclear, Biological, and Chemical (NBC) attack detection and reconnaissance.

As an example for military applications:

- **Sniper Detection System:** The Sniper location detection application (Boomerang sniper detection system) (Karlof & Wanger, 2003) has been developed for accurate sniper location detection by pinpointing small-arms fire from the shooter. This system has been used by the military, law-enforcement agencies, and municipalities, and has been realized through two distinct architectures (Akyildiz & Canvuran, 2010).

The counter sniper system (Figure 3.5) consists of a matrix of microphones that can be mounted on a vehicle or worn by a soldier. The system uses passive acoustic sensors to detect incoming fire. The detected audio from the microphones is processed to estimate the relative position of the shooter. The Boomerang system is helpful in urban settings, where a soldier or a vehicle can be subjected to fire from any location. The vehicle-mounted sensors can detect the shooter even while moving.

The network consists of Mica2 nodes equipped with a custom-made sensor board. The sensor board is embedded with a high-power DSP to provide real-time detection, classification, and correlation of acoustic events. The moment a shot has been detected by several sensors, the time and location information of these sensors is used to determine the trajectory of the bullet and estimate the location of the shooter. As a result, the distributed system improves the accuracy of the centralized system. This system is suitable for law-enforcement agencies and municipalities to provide protection during events such as speeches.

Figure 3.5 Boomerang sniper detection system
(Akyildiz & Canvuran, 2010)

3.2.4 Health Applications

The usage of WSNs for biomedical applications has been possible especially after the developments in implanted biomedical devices and smart integrated sensors (Akyildiz & Canvuran, 2010). Moreover, merging wireless sensor technology into health and medicine applications has made life much easier for doctors, disabled people and patients. Where, it can be used in the provision of interfaces for the disabled, integrated patient monitoring, diagnostics, drug administration in hospitals, tele-monitoring of human physiological data, and tracking and monitoring doctors and patients inside a hospital. Furthermore, high-quality health care services will get closer to the patients. Since WSNs make diagnosis and consultancy processes regardless of location and transition automatically from one network in a clinic to the other installed in patient's home. As a result, health applications are critical, because vital events of humans must be monitored (Verdone et al, 2007). Finally, the more researches and progresses in this field the better quality of life can be achieved and medical cost can be reduced. The following scenarios are related to this area:

● Night shift assistant
● Patient Monitoring

- **Night shift assistant:** WSNs, in night shift assistant application have supported the health care work in a standard situation in which many patients have to be managed by drastically reduced staff (e-SENSE, 2006) and it is, also, support the nursing homes. For example, a nurse that has to work night shift and responsible for a whole ward by him/herself. All patients can be equipped with sensors that measure their vital functions while at the same time keeping the patients mobile so they can, depending on their health condition, move freely without additional equipment. The patients' data can be observed on multi screens from central room, moreover, an alarm sounds that pays special attention to the data of a specific patient in case of a dangerous change in the patient ' s condition (Verdone et al, 2007).

- **Patient Monitoring:** At Harvard University, the CodeBlue project focuses on wearable sensors that monitor vital signs of patients throughout their daily lives (Akyildiz & Canvuran, 2010). Sensor boards for MicaZ and Telos motes with pulse oximeter, electrocardiograph (EKG), and electromyograph (EMG) circuitry have been designed (figure 3.6 taken from Malan, Fulford-Jones, Welsh, & Moulton, (2004)). Different situation like, pulse rate, blood oxygen saturation, electrical activities of the heart, patient movements, and muscular activity can be monitored continuously. Medical personnel can monitor patients through a PDA, where the project software, the CodeBlue software, platform enable these nodes to be operated in a networked setting.

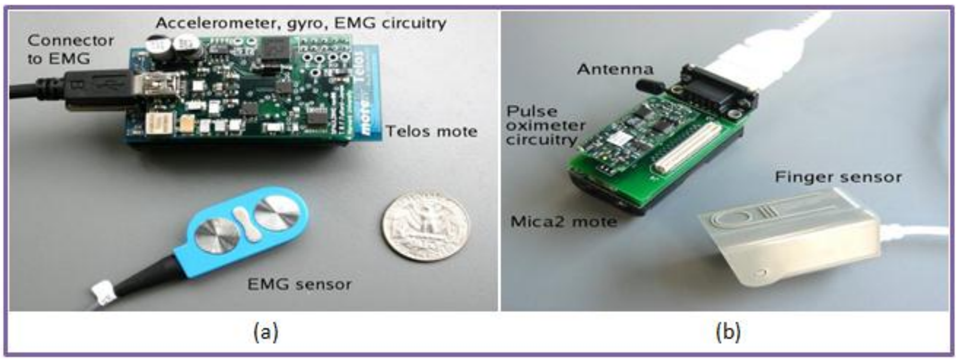

Figure 3.6 Medical sensor nodes used in CodeBlue:
(a) Telos mote with EMG sensor and (b) Mica2 mote with pulse oximeter sensor

Figure 3.7 shows the architecture of the CodeBlue network in a hospital setting, where sensor motes with various sensors are attached to patients for monitoring. Moreover, as an infrastructure, additional motes are deployed inside the hospital to provide message delivery to/from the motes. The network can be accessible through either PDAs or computers (Akyildiz & Canvuran, 2010). Mechanism of *publish/subscribe* is used in this network, where the sensor motes publish the available data and the medical personnel subscribe to this data based on certain conditions. Each node in the network is informed about the best path to a node that publishes a data. Accordingly, to adapt to the changes in the wireless channel route discovery between nodes is performed periodically through limited flooding.

Locating doctors, nurses, patients, and important equipment inside a hospital done by a localization service (Akyildiz & Canvuran, 2010), that provided by CodeBlue project, through a set of nodes that periodically broadcast beacon messages and mobile node can estimates its location according to the received RF signal signature from these beacon nodes a mobile node can estimates its location. Finally, the medical personnel can easily utilize the system through the provided graphical user interface (GUI) to the users.

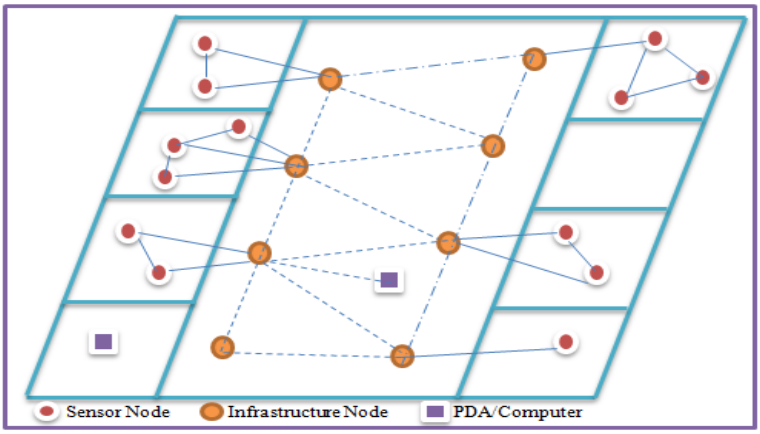

Figure 3.7 Code Blue architecture in a hospital setting
(redrawn from (Akyildiz & Canvuran, 2010))

3.2.5 Home Applications

Using smart sensors nodes and actuators in appliances such as vacuum cleaners, microwave ovens, refrigerators, and DVD players (Petriu, Georganas, Petriu, Makrakis, & Groza 2000) as well as water monitoring systems (Kim, Schmid, Charbiwala, Friedman, & Srivastava, 2008) after the advanced technology, have become possible. The sensor nodes can interact with each other inside domestic devices and with the external network via the Internet or satellite. In addition, they give a chance to end-users to more easily manage home devices both locally and remotely (Akyildiz & Canvuran, 2010). Furthermore, WSNs enable the interconnection of various devices at residential places with convenient control of various applications at home. The following application is a scenario related to this area:

- Water Monitoring
- Smart home

- **Water Monitoring:** The Nonintrusive Autonomous Water Monitoring System (NAWMS) is one of the applications that have recently been developed using WSNs (Kim et al, 2008) and aims to localize the wastage in water usage and inform residents about more efficient usage. , it is not easy to determine the individual sources that contribute to that total, where the water utility companies only provide total water usage in a house. So by using distributed WSN, the water usage in each pipe of the house's plumbing system can be monitored at a low cost.

The operating principle of NAWMS is simply based on the capability of estimating the water flow in a particular pipe by measuring the vibrations of that pipe. Moreover, to measure the vibrations through accelerometers wireless sensor nodes are attached to the water pipes. As the relationship between vibration and water flow is nonlinear, to determine the optimal set of parameters that relate acceleration information to water flow, each sensor node needs to be calibrated that performed automatically with the help of the main water meter.

The architecture of NAWMS consists of three types of components (figure 3.8). A wireless sensor node that attached to the main water meter works as a collector to actual water flow information and communicate it to the rest of the network and serves as a ground truth for the rest of the system. Vibration sensors which are the second component of this system are installed in each pipe to monitor vibrations (Akyildiz & Canvuran, 2010). The distributed vibration information is then sent to the central computation node that automatically calibrates the sensors and determines the water usage associated with each pipe. Calibration is performed through an online optimization algorithm, which is based on the fact that the sum of the water flows at each non-calibrated sensor should be equal to that of the water meter, which is known to be accurate. Finally, to estimate the flow rate at spatially distributed locations, the vibrations are used in a microscopic flow model and the system provides real-time water usage information at different locations of the water pipe system, which can be utilized to improve the efficiency of homes in the future.

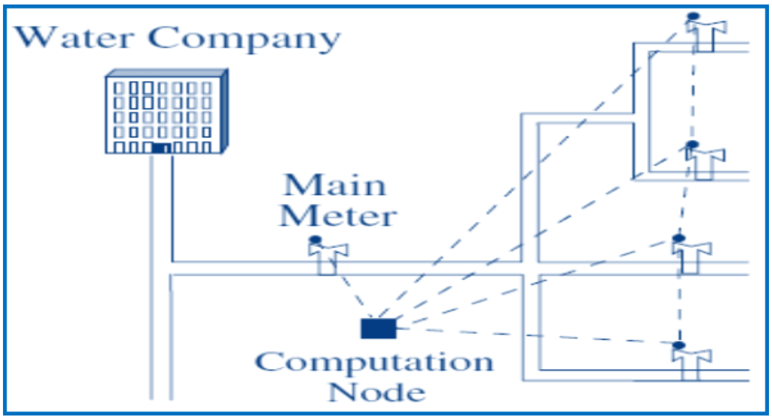

Figure 3.8 Non-intrusive Autonomous Water Monitoring System (NAWMS) Architecture

- **Smart home:** The ability to turn lights on and off remotely, monitor a sleeping baby without being in the room and having a fresh cup of hot coffee in the kitchen for breakfast was an amazing idea for home automation (Verdone et al, 2007). Smart homes are capable to acquire and apply knowledge about human surroundings and adapt to improve human experience (figure 3.9). Intelligent decisions in an automated manner can be made through the computing and communication capabilities, interaction with the information web, and its advanced electronics that enable early detection of possible problems and emergency situations.

A smart home works in a different way of a normal home, which is operated in a way that is more useful and more appropriate for the people living inside. Installed communication infrastructure allows various devices and systems in the home to communicate with each other. Adaptive control of home environment such as heating, lighting and ventilation can be provided and by increasing self-control and self-fulfillment a better quality of life can be achieved. As a result, smart home provide easy living and improve the social environment.

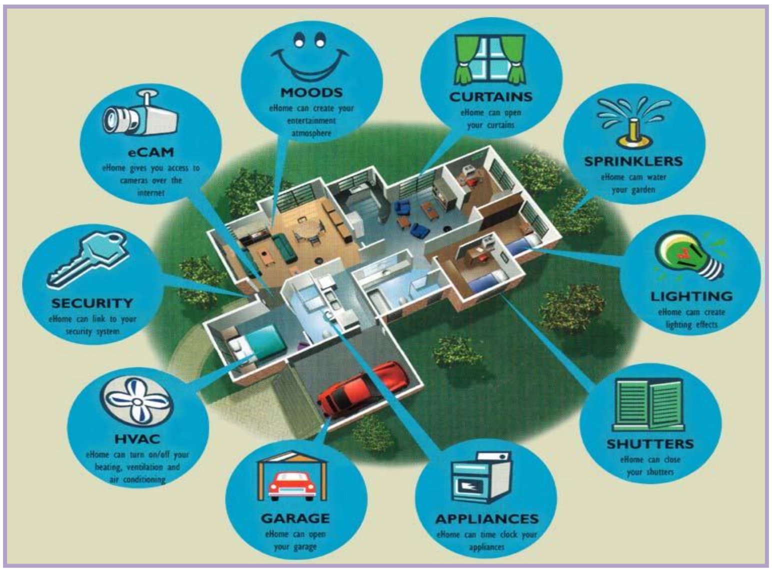

Figure 3.9 Smart home applications

3.2.6 Industrial Applications

In industrial fields such as industrial sensing and control applications, before, networks of wired sensors have been used for a long time, but the cost and the limitations of wired sensor network have limited the applicability of these systems. Instead WSNs have been used, which are a promising alternative solution for these systems due to their ease of deployment, high granularity, and high accuracy provided through battery-powered wireless communication units. The following scenarios are related to this area (Verdone et al, 2007):

- Smart factory
- Structural Health Monitoring
- Target tracking (Logistics)
- CORTEX (Transportation)

- **Smart factory:** The person who is responsible of maintenance of equipment and quality control in a factory (e-SENSE, 2006), for example food processing factory, he is using a WSN-based system in order to control the processes in the factory. Information like temperature, humidity, vibrations, lubrication, substance (e.g., moisture sensors) and other relevant parameters of a machine are collected through the sensors that installed on the machines. Reaching to the gateway destination where data from the network is gathered and processed done through communicating sensor node with each other. To having access to the instantaneous values of the sensors the person who is in charge is equipped with a remote monitoring, remote control, and data exchange enabled mobile device. When there is any abnormally situation, for example, when sensors could detect that the fruit press machine overheats and that the level of lubrication oil is low, an alert is sent to the mobile device (to the person) with the information about the thresholds and the location of the machine.

- **Structural Health Monitoring:** Structural health monitoring (SHM) applications that use WSNs can track, through distributed sensors, the spatio-temporal patterns of vibrations induced throughout the structures (Chintalapudi et al, 2006). Moreover, potential damage can be localized and its extent can be estimated in almost real time. In order to address the limitations of existing SHM techniques which depend on either periodic visual inspections or expensive wired data acquisition systems, Wisden has been developed (Paek, Chintalapudi, Cafferey, Govindan, & Masri, 2005). The actuators which networked with sensors apply forced excitation to the structure and detect the effect of these excitations. Through the distributed WSN the challenges of intolerance to data loss, strict synchronization requirements, and large volumes of raw data are addressed. The system has been deployed in a SHM testbed for development as well as in an abandoned building for evaluation.

- **Target tracking:** By using WSNs a product can easily be followed from production setup till it is delivered to the end user. However, this process may cover a large geographical area and many different entities to establish communication between entities involved in the application. Accordingly, mobility and localization of the components are basic requirements of the applications in this category. In this application scenario (Target tracking), the aim is to monitor the trolleys for baggage in a railway station (CRUISE, 2006), where the trolleys are located in some fixed locations distributed in the area. Sometimes, the passengers use these trolleys and they do not return them to their appropriate locations, furthermore, some of them could be taken outside the railway station. The problem that has to be solved is to locate these trolleys. So each trolley has been equipped with active Radio frequency identification (RFID) tag and each fixed location and entrance or exit with a sensor node equipped with an RFID reader. Then, these sensor nodes periodically collect the position information coming from the RFID tags, and by organizing themselves in a WSN they send all the position information, to the master (final user) of the network or to the central office of the station.

- **CORTEX:** Obviously, all applications used in transportation aim to provide people with more comfortable and safer transportation conditions. These applications could offer a variety of services like governmental or commercial services and aim primarily to obtain an autonomous transportation system. The localization is the basic issue of the mobile components. Real time, synchronization of the components and the end-to-end delay of the whole system are critical for such systems. There is different application scenarios related to transportation, for example, car control project (CORTEX) (Sihavaran et al, 2004). Where cars may communicate with each other to move safely on the road, reduce traffic conditions and reach their destinations. For reaching the destination, the implementation system will automatically select the best route according to desired time, distance, current and predicted traffic, weather conditions, and any other information that will be necessary for the purpose. Accordingly, if there are some obstacles or cars are approaching other cars, they slow down automatically or speed up in the absence cars and obstacles. Finally, the cars automatically obey rules of the traffic lights.

3.2.7 Other Commercial Applications

WSNs have a wide range of commercial applications and this is clear where WSNs have been used nearly in all areas of daily life, offer valuable application and variety of services in both governmental and commercial (Verdone et al, 2007). Bellow some of these commercial applications:

- Wireless automatic meter reading (AMR)
- Heating, ventilating, and air-conditioning (HVAC)
- In some Industrial applications

Wireless automatic meter reading (AMR): Wireless collection of utility meter data (electricity, water, gas) has been one of the fastest growing markets for short-range radio devices. Accordingly, it is a very cost-efficient way of gathering consumption data for the billing system (Swami et al., (2007), Chakrabarti & Seberry, (2006)). Low-cost, low-power radio chips and transceivers for wireless AMR applications can be produced by *Chipcon* (Wilson, 2005).

- ***Heating, ventilating, and air-conditioning (HVAC):*** WSNs have an important impact on HVAC applications especially in commercial buildings, where controlling multiple spaces or rooms by a single HVAC is common in such buildings. Accordingly, commonly, a single sensor in one of the rooms is used to control such systems. However, low-cost WSNs could be used instead of single sensor and in such network there is at least one sensor per room. Finally, for residential and light commercial control and status reading applications such as meter reading, lighting, and appliance control *ZenSys* produces wireless RF-based communications technology designed to this purpose.

- **In some Industrial applications:** Many industrial applications use products of *Sensicast* such as *H900 Sensor Network Platform,* which is a wireless mesh networking system (Rabaey et al, 2000). Furthermore, for real-time monitoring *SYS Technologies* produces such systems to monitor wide variety of remote industrial applications like wastewater, oil and gas, utilities, and railroads (Karl & Willig, 2005). Finally, security systems provided by *Soflinx* used for the real-time detection of hazardous explosive, nuclear, biological, and chemical warfare agents. Furthermore, a transportable security system has also been produced for the detection of the above hazardous agents.

CHAPTER FOUR
CLUSTERING AND ROUTING PROTOCOLS

4.1 Clustering Protocols

The high-density deployment of WSNs makes it different from other traditional networks (Akyildiz & Canvuran, 2010). It has accommodated with some advantages like coverage and connectivity on one hand, and disadvantages of collision and overhead in the other hand. As a result, the scalability is one of the major problems of high deployment WSNs protocols. Accordingly, the recent advances that witnessed in WSN made it interested in the applications that need high-deployment, i.e. environmental monitoring applications, where the sensor nodes are left unattended in order to report the parameter of interest like humidity, temperature, light, etc. (Younis, Krunz, & Ramasubramanian, 2006). Because of the difficulty of recharging node batteries in the case of such deployment, the energy efficiency became a major design goal in WSNs.

Some applications just need the aggregated value to be sent to the sink (or base station), so the sensors collaborate with each other to send more accurate information about their local locations. In order to help sensor nodes to aggregate data in efficient manner, the nodes themselves can be partitioned into smaller groups called clusters. A cluster consists of a number of nodes called *cluster members (CM)* and a coordinator called *Cluster Head (CH),* which aggregates the collected data by cluster members and this lead to more energy conservation. Cluster members communicate with each other and with respective CH locally. Furthermore, the CHs construct another layer of cluster by communicating with each other before reaching the sink or base station (BS). As a result, clustering has two-tier hierarchy; (i) lower tier formed by cluster members and (ii) upper tier formed by CHs. Figure 4.1 which is taken from (Younis et al, 2006) illustrates the clustering concept.

As the collected data by cluster members is aggregated by a respective CH, which in turn sends this data through other CHs to the sink, more energy is consumed by the CHs

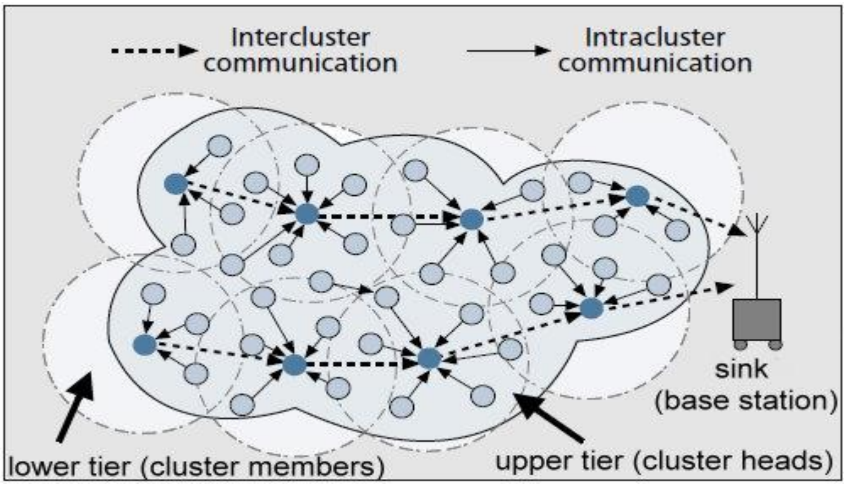

Figure 4.1 Clustering tiers and data flows

than the members. Moreover, the distance between one CH and the other is often longer than the distance between the cluster members and their respective CH. Therefore, in order to balance the energy consumption between the nodes, the network could be reclustered periodically to elect other sensor nodes (cluster members) to serve as a CH. Thus clustering, in addition to balancing the energy consumption among all the nodes, provides a better network throughput under high load by reducing the channel contention and packet collisions. Generally, the advantages of clustering protocols in WSNs are (Akyildiz & Canvuran, 2010):

- *Scalability:* Since the transmission numbers among the nodes are limited, the number of deployed nodes in the network could be high.

- *Collision Reduction*: Because the CHs work as a coordinator, number of nodes that access the channel is limited, where communication between cluster members and CH is local.

- *Energy Efficiency*: Because of the periodic reclustering in the network, the duties of CH, which consumes more energy, are distributed to the other nodes. This leads to less energy consumption.

- *Local Information*: Due to local communication between cluster members and respective CH, the situation of both local network and phenomenon are summarized at the CH (Younis & Fahmy, 2004).

• *Routing Backbone*: As the CH aggregates the collected data by its cluster members and sends it to the sink, means little route-thru traffic, efficient routing backbone in the network can be built.

4.1.1 Classifying Clustering Techniques

We can classify clustering techniques for WSNs based on network architectural and operation model and the objective of the node grouping process.

4.1.1.1 Network Model

WSNs applications have many architectural and design constraints. Some of these architectural parameters are (Abbasi & Younis, 2007):

• *Network Dynamics*: Clustering in case of mobility; node mobility, can be very challenging issue due to dynamic changes of the node membership that cause the evolution of clusters over time.

• *In-Network Data Processing:* Due to high-deployment, considerable redundant data can be generated by the nodes. So, to reduce the number of transmission data aggregation eliminates duplications or combines data by using min, max, and average (Krishnamachari, Estrin, & Wicker, 2002). Participating of the sensor nodes in the data reduction reduces the power consumed in the communication and the data aggregation and leads to traffic optimization.

• *Node Deployment and Capabilities:* According to the application, the sensor nodes deployment either deterministic or self-organizing. In deterministic situation clustering is not necessary but in self-organizing, where the nodes are scattered over the field randomly, it is necessary. Moreover, position of the base station or CH is important in terms of energy and performance.

In homogenous WSN; sensors are equal in their capability of computation, communication and power, CHs are chosen from the deployed nodes and to reduce their energy consumption they are excluded from duty of sensing. Whereas, in heterogeneous WSNs; sensor nodes are does not have same capability, clustering process has more constrains because some nodes maybe designed with distinct capabilities. Therefore, CH election may be done among subset of nodes and others are avoided to be chosen as CH.

4.1.1.2 Clustering Objectives

Often application requirements assign the clustering objectives. Below some important objectives for network clustering:

• *Load Balancing*: As CHs duties are more than the cluster members, it is important to recluster the network periodically in order to distribute the CHs duties to other nodes. Furthermore, choosing equal-sized clusters are also important in terms of increasing network life times.

• *Fault-Tolerance*: Nodes in application of harsh environment are exposed to failure. Therefore, in order to tolerate these failures of CHs it is good to recluster the network periodically. Moreover, sometimes the CHs have important sensor data, so making backup CHs is also an efficient way to recovery from CH failure.

• *Increased Connectivity and Reduced Delay:* Since the CH aggregates the collected data and sends it through other CHs to the base station the inter-CH connectivity is an important requirement. However when the latency is important, intra-cluster connectivity becomes an objective of the design.

• *Minimal Cluster Count:* Some CHs could be expensive, for example they could be laptop computers, robots or maybe a mobile vehicle, and so the designer wills to employ such expensive and vulnerable nodes as less as possible. Moreover, the small number of deploying such nodes could be due to the complexity nature of these nodes.

• *Maximal Network Longevity:* Minimizing the energy consumption intra-cluster communication, placing cluster members near to their respective CH and distributing the load on the other clustering by reclustering all these lead to maximize the network life.

4.1.1.3 Taxonomy of Clustering Attributes

There are number of attributes used to categorize and differentiate clustering algorithms of WSNs. Some of these are below:

- ***Cluster Properties***: some characteristics related to the inner structure of the cluster and below some relevant attributes:

    - *Cluster Count:* according to the used application the number of CHs either are predetermined or chosen from the randomly deployed sensors and that's lead to variable number of clusters.

    - *Stability**:*** the clustering is said to be stable or adaptable, when the clusters count change the node's membership must change according to this change. Otherwise, it is said to be fixed non adaptable.

    - *Intra-Cluster Topology:* cluster members usually communicate with respective CH directly. However, when communication range is limited, multi-hop communication between cluster members and respective CH is required.

    - *Inter-CH Connectivity:* sometimes the CH is limited or unable to communicate over long distances. In this situation, the cluster scheme must guarantee the inter-CH connectivity. Accordingly, it must guarantee the connectivity between CHs and the base station also.

- ***Cluster-Head Capabilities:*** As mentioned earlier, the nodes capabilities vary from node to other. Below some of the CH node attributes:

    - *Mobility:* Stationary CH leads to stable cluster. However, sometimes, for a better performance, the CH could be mobile. This mobility causes the cluster members to change their memberships dynamically and the clusters to be maintained continuously.

- *Node types:* Some sensor nodes are designed to serve as CHs intentionally and others are designed with more/little capabilities of computation and communication resources.

- *Role:* CH is the aggregator of the collected data by its cluster members and sends it to the base station, sometimes is just send the collected data to the sink, and sometimes, based on the phenomena, like a base station or a sink take some decisions.

- ***Clustering Process:*** According to clustering schemes the coordination and the characteristics of the entire clustering could be different. Below some of the relevant attributes:

  - *Methodology*: the clustering could be formed in three manners: (i) distributed manner, (ii) centralized authority, or finally, (iii) hybrid; distributed (inter-CH coordination) and centralized (each CH form its own cluster).

  - *Objective of Node Grouping:* objectives like fault-tolerance, load balancing, network connectivity, etc. have been formed the clusters.
  - *Cluster-Head Selection:* selection done in two ways; predetermined and randomly from deployed nodes.
  - *Algorithm complexity*: The objective and the methodology play important role in producing clustering algorithms. Complexity and convergence of these algorithms are determined by the number of CHs and cluster members.

4.1.2 Clustering Algorithms for WSNs

Clustering is an affective procedure in high-scale node deployments. Below and under two groups; variable and constant convergence time, are different types of clustering algorithms. Not to forget being the scalability is the main objective of the clustering:

4.1.2.1 Variable Convergence Time Algorithms

Time plays important role in the convergence of clustering algorithms, so it is practical to implement algorithms like LCA and RCC in the networks that have small number of nodes. However, other algorithms are suitable for networks with large number of nodes. Furthermore, this type (variable convergence time algorithms), provide more control in clustering properties than the second type (constant convergence time algorithms). Bellow some these algorithms:

- *Linked cluster algorithm (LCA)***:** This clustering algorithm form such a backbone network the cluster members can connect to while they are moving and its objective to provide direct connection between the CHs and their relative cluster members. It works in this way:

1. Each node broadcast it's ID and listens to other nodes.

2. A node broadcasts the set of heard neighbors, and in this way each node knows its 1-hop and 2-hop neighbors.

3. The node with highest ID became a CH and the adjacent nodes with the lower ID became its cluster members.

- *Random competition based clustering (RCC):* RCC supports mobile nodes and it is uses the First Declaration Wins rule; any first node claims to be CH could govern the other nodes within its radio coverage, where the nodes that hear the CH claim are join it as cluster members. Cluster forming done in this way:

1. Every CH in the network broadcast a CH claim packet periodically
2. To avoid conflicting of concurrent broadcast could be caused within the time of sending and receiving CH claims, RCC to judge between these broadcasts uses a random timer and node ID.

3. Every node before broadcasting its CH claim, it reset its random time value. If it receives a CH claim from another node, it will stop sending its claim and compares the IDs, and the node with a lower ID becomes CH.

This algorithms is practical when no topology knowledge or no access to global IDs. However, it is not when the clusters have their CHs in 1-hop range because the cluster will fail and the constructing process will restart.

- ***Hierarchical control clustering:*** the main objective of this algorithm is forming a multi-tier hierarchical clustering (Banerjee & Khuller, 2001). Cluster's properties like cluster size and degree of overlapping are considered in this algorithm. Figure 4.2 shows the concept of hierarchy of clusters. Any node in the network could start the cluster constructing but the node with the less ID has the priority. However, at the same time multiple nodes could start the process. In this case the algorithm runs through two phases; Tree discovery and Cluster formation.

(i) Tree discovery phase:

- Distributed formation of a Breadth-First-Search (BFS) tree rooted at the initiator node.
- Each node, n, broadcast a signal contain information about its shortest hop-distance to the root, r, every units of time, t.
- A node v which is neighbor of n it will choose n as its parent and if the route through n is shorter it will update its hop-distance to the root.

(ii) *The cluster formation phase:*

- This phase starts when a sub-tree on a node exceeds the size parameter, k.
- If the size of the node sub-tree <2k, the node will form a single cluster. Otherwise it will form multi clusters.

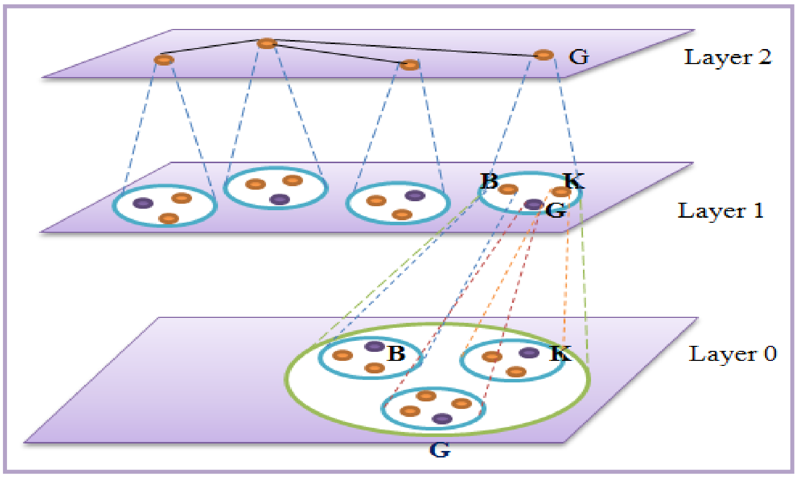

Figure 4.2 Three Layers Cluster Hierarchy
(redrawn form (Banerjee & Khuller, 2001)).

- ***Energy Efficient Hierarchical Clustering (EEHC):*** This algorithm has been proposed by (Bandyopadhyay & Coyle, 2003). Maximizing the network lifetime was its main objective. The CHs aggregate the collected data by their corresponding cluster members and send it to the base station based on two stages (i) initial and (ii) extended stage.

(i) Initial stage: each node elect itself as CH (called volunteer) and announce this to the other nodes that within its communication ranges with probability, p. Any node, which is not CH, within k hop range of CH receives this announcement becomes a member of this CH. The node that does not receive this announcement within time, t, and it is neither CH nor a member, means it is not within k hops of volunteer CHs, become a forced CH.

(ii) Extended stage: in this stage the process extended to include multi-level clustering likes that shown in the figure 4.2, where the clustering process is repeated to construct extra tier contains the CHs. And transmitting between these tiers done in a way that: Cluster members send the collected data to the layer 1 CHs, and the CHs at the layer-1 send the aggregated data to layer 2 and so on tell the base station.

4.1.2.2 Constant Convergence Time Algorithms

Constant convergence time algorithms, regardless the number of nodes, are converge in a fixed number of iteration (Abbasi & Younis, 2007).

- *Low Energy Adaptive Clustering Hierarchy (LEACH)***:** It is one of the most popular clustering algorithms in WSNs (Heinzelman, Chandrakasan, & Balakrishnan, 2002). The cluster is constructed depending on the strength of the received signal, by using distributed algorithm. Collecting, processing and aggregation of the data are done locally in the cluster. Moreover, the data use the CHs as a route to reach the base station. Processes of the forming cluster are:

1. With a probability of *p,* a node elects, announces, itself as CH.
2. The cluster is formed by choosing each non-CH the CH that can reach by using the least communication energy.

To balance the load (energy) between the nodes, the elections between the nodes to be a CH is done periodically and in this way:

1. Each node get a random number, T, between 0 and 1
2. A node becomes a CH if its number less than T(*i*) where:

$$T(i) = \begin{cases} \frac{p}{1-p*(r \bmod \frac{1}{p})} & \text{If } i \in G \\ 0 & \text{Otherwise} \end{cases} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.1)$$

Where, according to (Abbasi & Younis, 2007):

p is the desired percentage of CH nodes in the sensor population.

i is the current round number.

G is the set of nodes that have not been CHs in the last 1/p rounds.

This algorithm is not recommended in networks setup in large area, because even the nodes with very low energy can be a CH; where CH must be able to communicate over long distances.

• *TL-LEACH:* Two-Level Hierarchy LEACH (or TLLEACH) is extension of the LEACH algorithm (Loscri, Morabito, & Marano, 2005). In addition to the simple nodes, there are two levels of CHs: primary and secondary. In each cluster the primary CH communicate with the secondaries, which in turns communicate with the nodes in there sub-cluster as shown in the figure (4.3). Like LEACH, It uses the same mechanism to select the primary and secondary CHs. However, probability to be primary CH is less than to be secondary. Data from the source to the sink can travel in two steps (Loscri et al, 2005):

1) Secondary nodes collect data from nodes in their respective clusters.
2) Primary nodes collect data from their respective secondary clusters.

Due to the nature of the two-level structure of TL-LEACH, amount of the nodes that need to transmit to the base station is reduced. As a result, the total energy consumption is reduced.

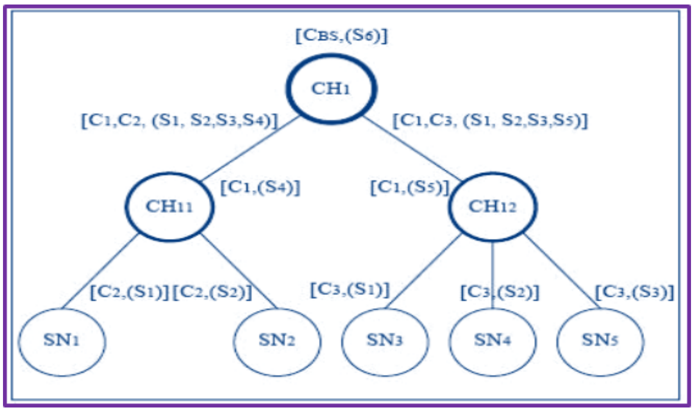

Figure 4.3 TL-LEACH in Clustering

• *Energy Efficient Clustering Scheme (or EECS):* In this algorithm the CH candidates compete to be a CH. Competition involves that each candidate broadcast its residual energy to the other candidates, and the node with more residual energy becomes a CH. EECS differs from LEACH in way of cluster constructing. The second constructs clusters based on the minimum distance between the nodes and their respective CH. However, EECS form the cluster based on the distance between the cluster and the base station (Ye, Li, Chen, & Wu, 2007).

This algorithm addresses the problem that transmission of clusters at the long distance from the base station requires more energy than the closer ones. As a result, energy distribution through the network will be improved and network lifetime will be extended.

- *PEGASIS:* Power-Efficient GAthering in Sensor Information Systems (Lindsey & Raghavendra, 2002) is a data-gathering algorithm uses one CH to transmit packets to the base station for each round, see figure (4.5). However, in LEACH all the CHs participate in the transmission. Idea of this algorithm is that *"if nodes form a chain from source to sink, only 1 node in any given transmission time-frame will be transmitting to the base station"* (Dechene, El Jardali, Luccini, & Sauer, 2006). In this algorithm at every node in the WSN data-fusion could occur and a node need less transmission range to transfer data than in LEACH, which leads to energy improvement.

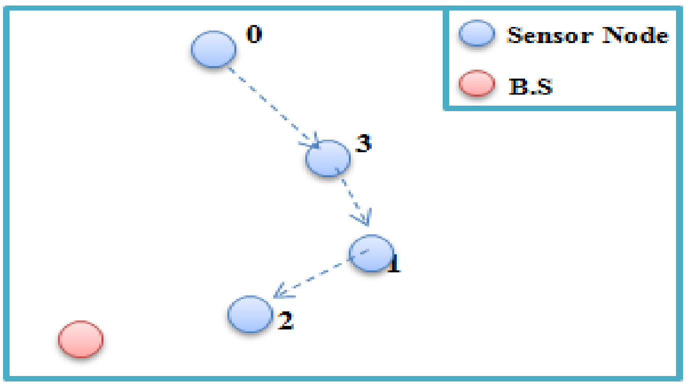

Figure 4.4 PEGASIS in Clustering

- *Fast Local Clustering service (FLOC):* equal size and minimum over-lap clusters can be achieved in this algorithm, where it is using distributed technique. Nodes are classified into two bands (Abbasi & Younis, 2007):
  (i) Inner band (i-band); favorite band since there is no obstacle.
  (ii) Outer band (o-band); not favorite and massages maybe lost.

The algorithm processes flow as shown in the figure 4.3 (Redrawn from Demirbaş, Arora, & Mittal, (2004) :

- A node candidate itself as CH if it does not receive an invitation from other CH and broadcast a candidacy message (Transmission 1).
- The candidate CH becomes CH and invites other to join its cluster (Transmission 4) or if it receives a conflict message it will join another cluster as o-band node (Transmission 3).
- An idle node if it receives an invitation from a near CH it will join it as i-band (Transmission 2) otherwise it will join the cluster as o-band (Transmission 5). However, if it receives an invitation from a near CH it will change its membership to the better; to i-band (Transmission 6).

FLOC has many advantages; for example, convergence time is constant does not depend on the network size, self-healing where the nodes can change their membership from o-band to i-band, and finally, new nodes either join an existing cluster or construct a new cluster by inviting o-band nodes from neighbor clusters

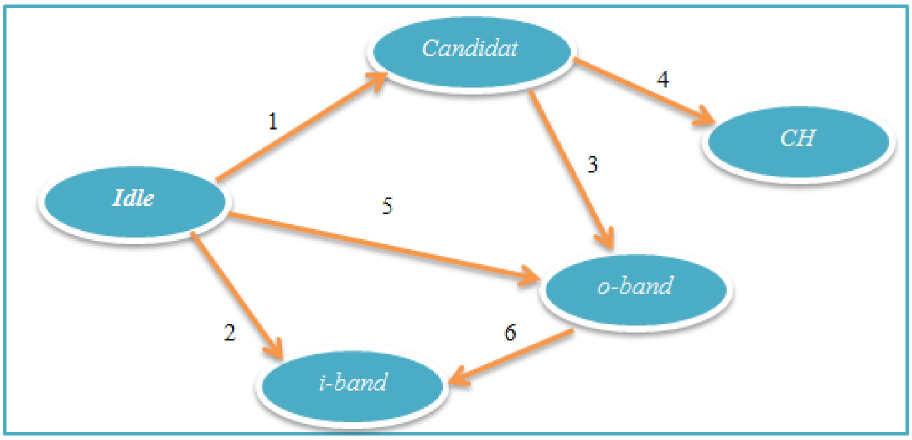

Figure 4.5 State transitions for the FLOC clustering algorithm

- ***Hybrid Energy-Efficient Distributed Clustering (HEED):*** It is, like LEACH, uses distributed technique. However, it differs from LEACH in term of choosing the CH; where HEED chooses nodes with high energy to serve as CH and LEACH chooses it randomly. HEED has three main characteristics (Abbasi & Younis, 2007):

- Two nodes within each other's radio rang to be a CH is almost non-existent. This shows the well distributing.
- Nodes are different in their energy consumption.
- The probability of CH selection can be adjusted to ensure inter-CH connectivity.

Each node in HEED can communicate with relative CH directly. The algorithm processes can be divided into three phases:

1. ***Initialization phase*:** initial percentage of CHs,"Cprob", is set by the algorithm in order to reduce the CH announcements between the sensors. Probability of being CH,"CHprob", is set by each sensor and this way:

CHprob = Cprob * Eresidual/Emax,

Where:

Eresidual: is the current energy in the sensor

Emax: is the maximum energy, which corresponds to a fully charged battery.

CHprob never be smaller than certain threshold, pmin, (inversely proportional to Emax).

2. ***Repetition phase:*** in order to join a CH and with a minimum transmission range, all the sensors goes through several iterations. Otherwise the sensor announces itself as CH. The iterations of this phase will continue till CHprob reach 1, where each sensor doubles its CHprob before going to the next iteration. Finally, according to CHprob value there are two types of CH status:

- *Temporary status:* CHprob less than 1. When it finds lower cost (less communication range) CH, it will changes it status to a cluster member.
- *Permanente (final) status:* CHprob reach 1.

3. ***Finalization phase:*** in this phase each sensor either joins a least cost CH as member or announces itself as CH (orphaned node). However, in (Huang & Wu, 2005) a new version of HEED has been modified to re-execute the algorithm for those orphaned nodes. As a result, this modification has decreased the number of

CHs which in turn reduces size of the routing tree in inter-CH communication resulting in limited data collection delay.

4.2 Routing Protocols

Characteristics of WSN that distinguish it from other wireless networks make the routing in WSNs very challenging. So, because of the large number of sensor nodes and being the data more important than knowing the source sender, traditional routing protocols may not be applied to these networks. Moreover, in most applications the data that need to be send to the base station flow from multiple sources. Due to energy, storage capacities and processing constrains of sensor nodes, they need careful resource management. Furthermore, according to the application, the design requirements of WSN may be change (Verdone et al, 2007). For example, when designing the routing protocols, in military applications, the energy consumption is considerable factor, while in some monitoring applications the QoS must be taken into the account and the security in the application that need security services. Finally, because the data that sensed by many sensors in WSN, sometimes, they contain a lot of redundancy. So, In order to improve energy and bandwidth utilization, such redundancy must be exploited by routing protocols.

Network layer is one of the most important research topics in WSNs. Through WSNs development, many routing protocols and algorithms have been proposed. Although there are many challenges for this protocols and algorithms, solutions are exist. In order to discuss these solutions the routing protocols can be divided into four groups:

1 *Data-Centric Routing Protocols:* in this group of routing protocols instead of node ID attribute-based naming is required, where users are interested in querying attribute of the phenomenon rather than querying an individual node. Examples for this group are; Flooding, Gossiping, SPIN and Directed Diffusion.

2 *Hierarchical Routing Protocols*: the nodes in this group of routing protocols are divided into clusters. The cluster members interact with each other and with the Cluster-Head locally and they controlled by Cluster-Head. Some of hierarchical protocols are LEACH, PEGASIS, TEEN, APTEEN and EECR.

3 *Location based routing protocols*: in these routing protocols neighborhood information is by default inferred from the physical placement of the nodes. As a result, geographical protocols are scalable and because the next hop could be selected based on the local information they have low complexity. Some of these protocols are; MECN, SMECN and PRADA.

4 *QoS based routing protocols:* they, in addition to the energy metric, add additional metrics for constructing routes. Although this provides extra capability to WSNs, it increases the cost in terms of energy consumption and network life time (Akyildiz & Canvuran, 2010). Some of these routing protocols are; SAR and Min-Cost Path.

4.2.1 Challenges for Routing

Because of the effectiveness of communications and sensor networks features many challenges can face the efficient routing, where routing is one of the major problems in WSNs. However, to address this problem many solutions have been developed. Flowing are the major challenges:

Energy Consumption: As mentioned before that the limited energy, which is normally battery operated, is one of the main constrains in WSNs. So this constrain must be considered when developing efficient routing protocols that need to deliver the information from the source to the destination without any corruption and in the most energy-efficient manner. Routing in WSNs consumes energy for two main causes:

- *Neighborhood discovery:* Information like locations (geographical routing protocols) and content-based information (data-centric protocol) need to be exchanged between neighbor nodes. These exchanges consume energy through wireless communications.

- *Communication vs. computation*: Despite of being computation consumes less energy than communication (Akyildiz & Canvuran, 2010), to improve energy consumption computation must integrated with routing. For example, to decrease the traffic without corruption data from many nodes can be aggregated into single packet.

Scalability: In high-scale node deployment in order to improve the energy efficiency and decrease the load, routing protocols must support scalability and in-network combination; where most of the time high-level information (aggregated data) is accepted more than individual pieces of information from each sensor node.

Addressing: In large number sensor nodes deployment to decrease the overhead of routing, new routing protocols that do not require unique IDs for each node must be developed, Where information about a physical phenomenon to be read as a collective from multiple sensors are more usual and acceptable than to be read from individual sensors.

Robustness: Not as the Internet, where it depends on routers to deliver data in a multi-hop communication, WSNs depend on low cost components nodes. So these sensor nodes may face unexpected failures and result in non-operational nodes. As a result in order to ensure an efficient routing, routing protocols must provide robustness against node failure and should be effective and do not relay on a single packet that can be lost if a sensor node dies.

Topology: As mentioned before, deployment done in two ways: (i) predetermined and (ii) randomly. The second case is the usual one for WSNs, so the initial topology of the network is not aware by the individual nodes. Therefore, routing protocols must provide topology-awareness and should be adaptive to dynamic changes of the network topology accommodated with sink mobility and switching of the nodes between on and off.

Application: Routing protocols can differ from application to other, for example, monitoring applications use static routes and communication between the sensor and the sink is periodic. However, in event-based applications a route is generated whenever an event occurs. As a result, according to the applications, different routing techniques may be needed.

1.1.2 Taxonomy of Routing Protocols

In order to classify the different approaches to routing there are several ways. We are going to classify the routing protocols to four classes; Data-centric routing protocols, Hierarchical routing protocols, Location based routing protocols and finally QoS based routing protocols

1.1.2.2 Data-centric and Flat-Architecture Protocols

In large-scale nodes deployment it is difficult to assign IDs to each node, therefore in WSNs data centric routing protocols are more preferable than address-based routing protocols. This leads to energy efficiency because the routes are formed just when there is an interest. For example, it is more acceptable to say “the areas where the temperature is over 70 ◦F (21 ◦C)” than saying “the temperature read by a certain node.” Below are some protocols that apply data-centric routing mechanisms. However, the major disadvantage of data-centric routing protocols is the scalability and congestion in nodes near to the sink that because these protocols are based on flat topology.

- ***Flooding:*** it is the simplest routing algorithm developed for multi-hop networks. It does not need any information from its neighborhood and does not require costly topology maintenance and complex route discovery algorithms. A node simply broadcasts the received packet to all of its neighbors, means the packet will be flooded through the whole network, see figure 4.6, broadcast could be stopped when the packet reached its destination or maximum number of hops has been exceeded. However, flooding suffers from these disadvantages (Akyildiz & Canvuran, 2010):

- *Implosion:* Same node may receive the same duplicated message from its neighbors. For example, as shown in figure 4.7(a), when a node A sends a message *to node B through N nodes neighbors, the result is node B will receive N copies of* the same sent message.

- *Overlap:* When two nodes share an overlapping area, the result, neighbor nodes will receive same message from these two different nodes. See the red area in the figure 4.7(b).

- *Resource Blindness:* Flooding protocol does not care about energy resources, which is the most important issue that must be considered.

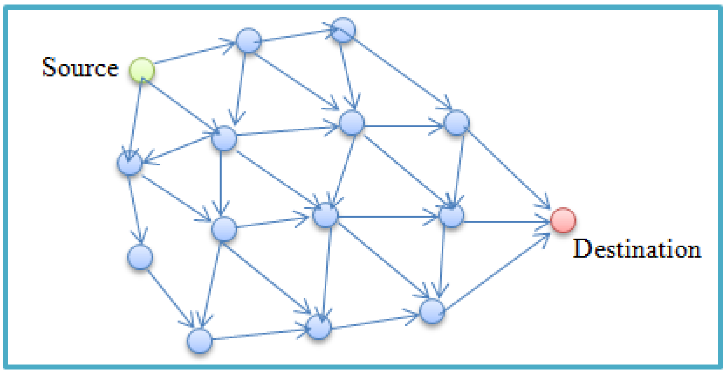

Figure 4.6 Flooding protocol

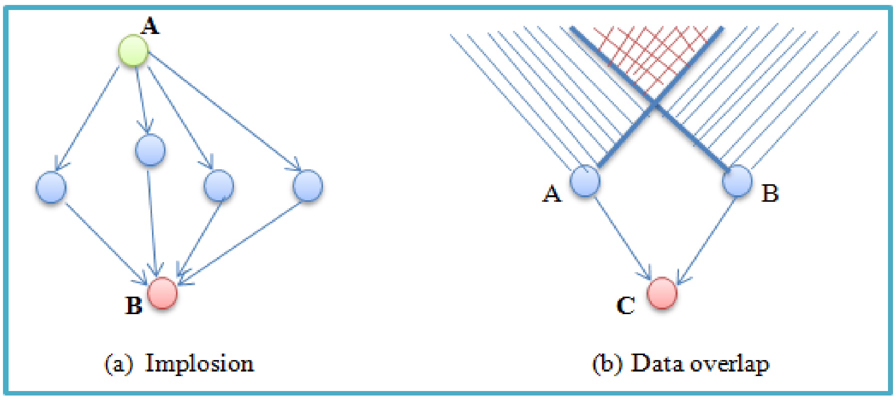

Figure 4.7 Flooding problems (redrawn from (Akyildiz & Canvuran, 2010))

- ***Gossiping:*** The implosion that faces the flooding could be avoided in the gossiping, where it chooses just one node to relay the packet to the destination and in this way: when a node has a packet to send it chooses a random node from its neighbor and forward it to that node. Once the selected node receives the packet it chooses, in turn, another random neighbor and forwards the packet and so on till destination see figure 4.8. Gossiping is less energy-consume than flooding. However, it suffers from latency; information propagates slowly, one node in each step. Despite of simplicity and inefficiency of flooding and gossiping, they could be used for specific functions, for example, during deployment phase and network initialization.

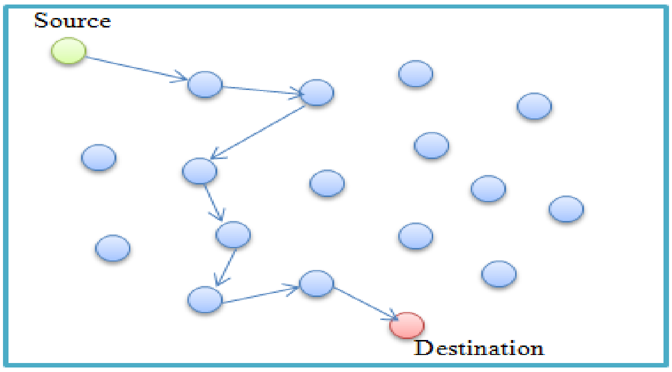

Figure 4.8 Gossiping protocol

- ***Sensor Protocols for Information via Negotiation (SPIN):*** Has been designed to overcome the inefficiency of flooding through two approaches: Firstly, the sensor nodes negotiate with each other through packets that describe the data. As a result the message is only sent to the concerned nodes, and secondly, energy-aware decisions are taken by monitoring each sensor node its energy resources. The algorithm consists of three types of messages, (i) advertisement (ADV), (ii) request (REQ) and (iii) DATA, and flow these three steps shown in the figure 4.9:

  Step 1: a node broadcasts an ADV packet, which is contains description of the DATA packet to be sent. ADV packet normally it is much smaller than DATA packet.

Step 2: the node/nodes which interests in this ADV packet will reply back with REQ message.

Step 3: finally, the DATA will be send to the requested node/nodes.

These three steps will be repeated until the DATA received by the destination sensor node/nodes. It is important to refer that the node with small energy than the threshold will not send REQ packet. However, to keep its residual energy, it is just receives the DATA packet. Finally, because of the handshake mechanism of the SPIN, it is higher latency than flooding.

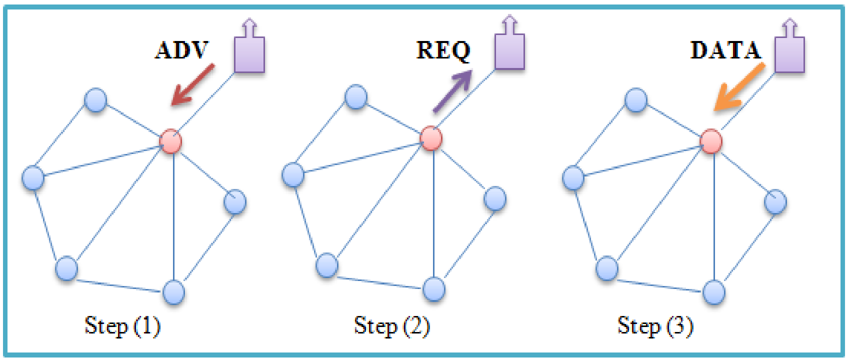

Figure 4.9 the SPIN protocol

- ***Directed Diffusion:*** The directed diffusion data aggregation paradigm has been developed to address the requirement of data flowing from the sink toward the sensors, i.e., when it requests particular information from these sensors (Intanagonwiwat, Govindan, & Estrin, 2000). In order to construct routes between the sink and the sensors, there are four phases: (1) interest propagation, (2) gradient setup, (3) reinforcement, and (4) data delivery.

  - *Interest propagation phase*: Directed diffusion starts when the sink sends an interest message to all sensors and this message will flood through the network. See figure 4.10(a).
  - *Gradient setup phase:* Each node receives the interest message and save it in an interest cache, which includes timestamp (shows the time of receiving the

interest), gradient (construct the reverse path to the sink), interval and duration (duration of storing and staying the interest in the cache). Then the node forwards, floods, the interest downstream to other neighbor nodes. Accordingly, the gradient is sent back to first node which sent the interest. As the interest spread through the network, the gradient will be constructed from the source back to the sink. See figure 4.10(b). The node becomes a source if its sensor readings match the interest and sends the data to the sink by using the interest's gradient path showed in figure 4.10(b).

- *Reinforcement phase:*_As the source can send the data through several paths back to the sink. The sink could reinforce a specific path (it is selected according to rules like best link quality or lowest delay) by resending the interest through specified node in that specified path. See figure 4.10(c).
- *Data delivery phase:* Finally, once the source node has been selected, the path between the source and the sink will be constructed as shown in the figure 4.10(d).

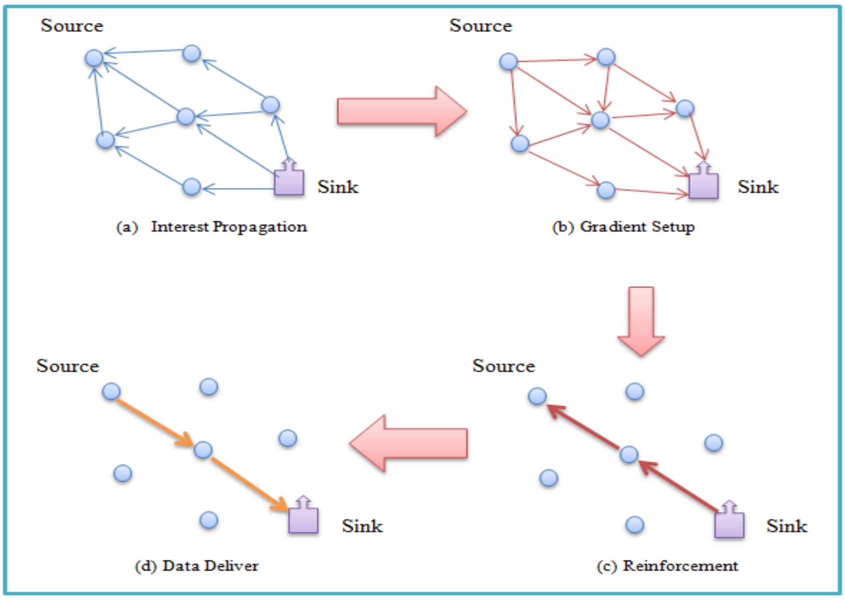

Figure 4.10 Operation of the directed diffusion protocol

1.1.2.3 Hierarchical Protocols

As shown above that the data-centric and flat-architecture routing protocols suffer from overload in the nodes near to the sink and this leads to fast die for this nodes and split the sink from WSN and end in unfair energy consumption and limit scalability routing protocols (Akyildiz & Canvuran, 2010).

To overcome the deficiencies of the flat-architecture routing protocols hierarchical architecture has been formed; Nodes grouped in clusters and controlled by a cluster head, See figure 4.11, and cluster heads among themselves could form another layer before reaching the sink. As a result the hierarchical routing protocols are scalable and less energy consumption than flat-architecture routing protocols.

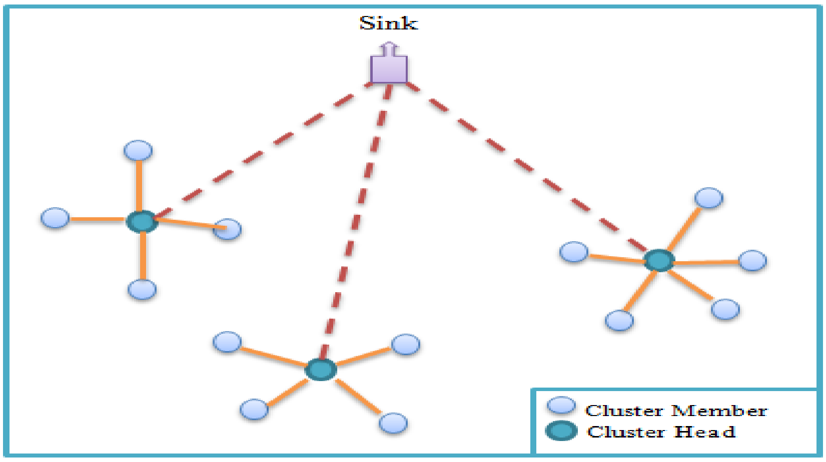

Figure 4.11 Hierarchical cluster-based architecture in WSNs

- ***LEACH:*** Objective of LEACH is to minimize energy consumption (Heinzelman, Chandrakasan, Balakrishnan, 2000), where it is dynamically chooses nodes to serve as cluster heads and others as cluster members. As the cluster head aggregates the observed data by the cluster members and sends it to the sink, it is consume more energy than cluster members. Therefore, the role of cluster head changes dynamically and thus the high energy consumption will be distributed among the sensor nodes in the network. LEACH is operated and controlled via rounds which in turn divided into two phases, (i) setup phase and (ii) steady stead phase, it is took more time than setup phase that is to minimize the overhead.

Setup phase: consists of three phases: (i) advertisement, (ii) cluster setup, and (iii) schedule creation. The cluster head is selected in the advertisement phase, where each sensor node broadcast itself as a cluster head. Then according to the Threshold equation (4.1) the selected cluster heads announce themselves as new CHs. Each node that receives this advertisement will join the CH with highest signal strength as a member. Moreover, to prevent collisions between the packets, LEACH uses CSMA-based random access scheme.

Steady state phase: in this phase the sensor nodes are able to sense and transmit data to the cluster heads and cluster heads, in turn, aggregate these data then send them to the sink. When this phase finishes the network goes into the setup phase again and enter another round; in this way the role of cluster head, the energy consumption, will equally distributed among sensor nodes.

Finally, LEACH improves the network life time, where only cluster heads are active during the steady stead phase. However, cluster members are active only during the setup phase and its allocated time slot. As a result, LEACH is less energy consumption, 4-8 factor (Akyildiz & Canvuran, 2010), than flat-architecture routing protocol.

- ***PEGASIS:*** Objective of PEGASIS is to overcome the overhead of cluster constructing in LEACH by constructing chains of nodes instead of clusters starting from the farthest node to the sink (Lindsey & Raghavendra, (2002), Lindsey, Raghavendra, & Sivalingam, (2001)), as shown in the figure 4.12. According to a greedy algorithm a node chooses the nearest node in the chain as its next hop and so on each node flows just its next and pervious neighbor in the chain. All nodes in the chain contribute in the data aggregation.

The aggregation done sequentially manner, where each sensor aggregates data from its neighbor till all the data are aggregated at chain leader, node 2 in the figure

5.6, for example the communication starts by passing the node 2 a token to node 0. Node 0 sends its data to node 1 then node 1 aggregates these data with its own and sends it to node 2. When node receive the aggregated packet from 1, it passes the token to the other end of the chain, node 6, and in tame way the aggregated data reaches node 2. Finally, node 2 will send the aggregated data to the sink in a single hop communication. PEGASIS less energy consumption than LEACH but it is has more latency because of the sequential aggregation

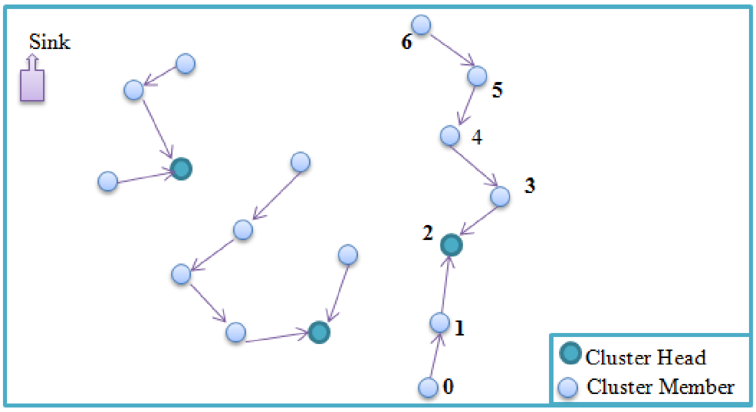

Figure 4.12 Chain structures of PEGASIS

- ***TEEN:*** Opposite LEACH and PEGASIS, where information is periodically transmitted from nodes to the sink, TEEN protocol provides event-based delivery i.e., the information is generated just when certain events occurs (Manjeshwar & Agrawal, 2001).

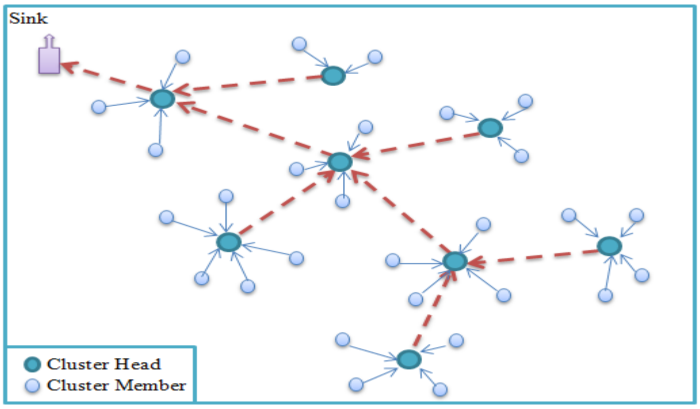

Figure 4.13 Hierarchical architecture of TEEN and APTEEN

As shown in figure 4.13, Sensor nodes are organized into multiple levels hierarchical architecture, where CHs aggregate the collected data by the sensors and transmit it to a higher level CH till the base station/sink. To achieve energy efficiency, role of CH is distributed to other nodes inside the cluster. Event-based communication in TEEN provided by two thresholds:

- *Hard threshold (HT):* if the sensed value exceeds the HT, the node sends its observed data to the CH.
- *Soft threshold (ST):* prevent or reduce redundancy. Sensor node will not send its data if the difference between consecutive observations does not exceed the ST.

- ***APTEEN:*** Adaptive threshold-sensitive energy-efficient sensor network (APTEEN) protocol has been developed as an advancement of TEEN because the last is not suitable for periodic reports (Manjeshwar, & Agrawal, 2002). Each node In APTEEN periodically sends its data to the CH. Furthermore, the HT and ST values could be controlled depending on frequency of data sending. As a result, both event-based and monitoring applications can be served.

- ***Energy-Efficient Clustering and Routing Protocol (EECR):*** The basic objective that must be taken into account when designing applications and protocols in WSNs is maximizing network lifetime by minimizing the energy consumption. So, within last few years many energy-efficient algorithms have been proposed for WSNs, like LEACH and the others. However, EECR differ from LEACH in a way that is chooses CHs with more residual energy (Liu & Yu, 2009). It achieves a well distribution of cluster heads. Moreover, in order to produce clusters of unequal sizes, EECN uses distance-based cluster formation method. Clusters that are away from Base Station are smaller sizes, to keep their energy for long distance transmissions.

Despite of the advantages of EECR, its main problem is using the lowest energy path for all transmissions between the source and destination. As a result, the node may exhaust its all energy along that path, hence lead to network split.

1.1.2.4 Geographical Routing Protocols

Location information in many applications, that its main interest is the physical phenomenon in the environment, is very important. Location information of a node can be provided by integrating GPS devices in the embedded board. Geographical routing protocols (Location-based protocols) use this location information to provide efficient routing (Akyildiz & Canvuran, 2010).

- *MECN:* For a given communication network the MECN (Minimum Energy Communication Network) drives an energy-efficiency sub-network. This sub-network minimizes the communication energy consumption in the network (Akyildiz & Canvuran, 2010). For a given network graph G', MECN drives the sub-graph G with the same number of vertices (nodes) but number of edges (connect two nodes) are smaller. See figure 4.14.

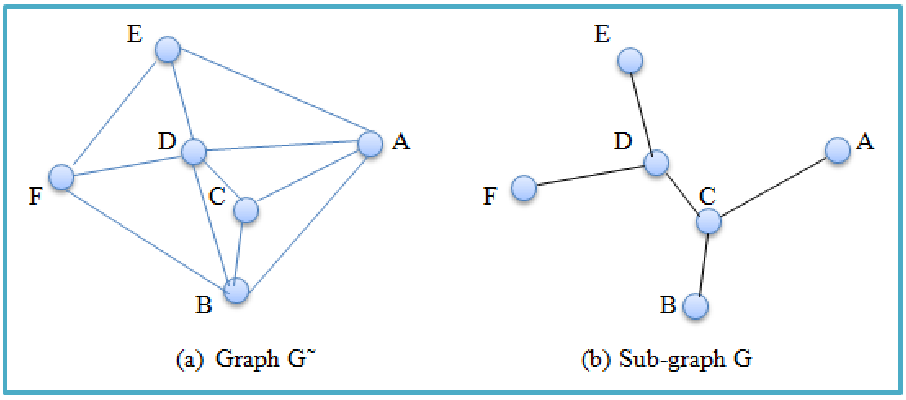

Figure 4.14 Sub-graph formation in MECN

In the Figure 4.14(a), although the node A can communicate directly with node B, it is more energy efficient to use a node C as a relay to reach node B. Moreover, direct communication between two nodes need high power communication. As a result, the sub-graph G, see figure 4.14(b), is less energy consumption than sub-graph G' (Akyildiz & Canvuran, 2010). For node A it is more energy efficient to use node C as a relay to reach other nodes that in relay region with node C.

- *SMECN:* Small MECN (SMECN) is a new algorithm proposed from MECN to improve the channel modeling in MECN (Li, & Halpern, 2001). Where it is contains fewer edges than MECN and follows minimum energy path to construct the sub-network.

- **PRADA (Probe-Based Distributed Protocol for Knowledge Range Adjustment):** Geographical routing protocols, as mentioned above, in order to choose the next hop they need information about the neighborhood. Moreover, neighborhood information on a node can give limited view of the network. However, an optimum path from source to destination can be found if the node has a global knowledge of the network (Akyildiz & Canvuran, 2010). As shown in figure 4.15, the solid line shows the optimum route when the nodes have global knowledge of the network where the dashed line shows the route when the nodes have knowledge about one-hop neighbor. As a result, a node must have topology knowledge to make an accurate forwarding decision. However it consumes more energy.

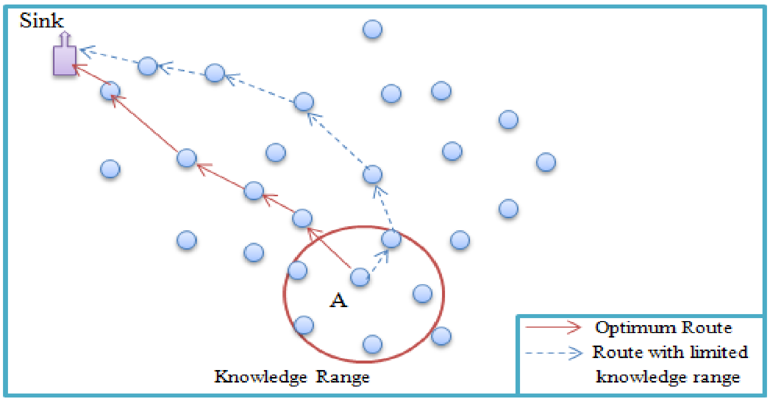

Figure 4.15 The impact of knowledge range on geographical routing.

PRADA, which is a distributed version of PTKF, is based on partial topology knowledge forwarding (PTKF), which is a centralized forwarding scheme aims to minimize the energy consumption for data communication and the cost for topology information. Based on a weighted shortest path algorithm each node and according to its topology knowledge construct a route. Moreover, each node in PRADA adjusts its knowledge range according to the feedback information it receives from neighbor nodes.

1.1.2.5 QoS-Based Protocols

Energy consumption is one of the most important metrics in WSNs but it is not the only one. Most of the pervious protocols focused on energy consumption. However, in some cases metrics like throughput and delay are maybe more important than energy consumption. As a result beside energy consumption, quality of service (QoS) requirements must be also considered when designing routing protocols (Akyildiz & Canvuran, 2010).

- ***SAR (Sequential Assignment Routing):*** SAR is one of the first routing protocols in WSNs that developed to provide QoS requirements, which is provide a table-driven multi-path approach (Sohrabi, Gao, Ailawadhi, & Pottie, 2000). Its objective is choosing a node located in a single hop from the sink and making this node as root node to multiple trees. Each tree grows away from the sink and avoids nodes with very low QoS and energy. Accordingly, multiple paths that connect any node in the network to the sink can be created (Akyildiz & Canvuran, 2010). Each node specifies two parameters for each path to the sink:

    • *Energy resources:* maximum number of packets that a node can send if it has exclusive use of the path.
    • *Additive QoS metric:* related to the energy and delay at each path.

As a result, according to the QoS of each path and the packet priority a node can choose its path from these multiple paths. When a node has a packet to send, it calculates a *weighted QoS metric* for the packet, which is a product of the QoS metric of the path and the priority level of the packet. Finally, for higher priority packets, paths with higher QoS are used.

- ***Minimum Cost Path Forwarding*****:** The minimum cost path forwarding protocol assigns a cost function, which is a combination of the delay, throughput and energy characteristics, to each link. According to this function, at each node *cost field* will be specified by the minimum cost forwarding algorithm. As a result, through this cost field that specifies the lowest cost next hop. The minimum cost path algorithm has two phases:

- *Cost field establishment phase*: determine the minimum cost between any node and the sink.
- *Cost path forwarding phase:* the sink broadcasts advertisement (ADV) message with an initial cost of 0 by updating the cost this message will be forwarded.

Each node *j* that receives an ADV message from node *I* will calculate its cost in this way:

$$L_i + C_{j,i} \tag{5.1}$$

Where

L_i is the cost of node *i* (= 0 for the sink)
$C_{j,i}$ is the cost from node *j* to node *i*.

Finally, Each node sets up a backoff timer (backoff timer help the node to update its cost to the sink by choosing the minimum cost node to the sink) proportional to its cost to node *i*, $C_{j,i}$, and broadcasts the ADV message.

CHAPTER FIVE
INTEGRATED COMPARISON OF ENERGY EFFICIEN ROUTING PROTOCOLS

As seen from previous chapters, Wireless Sensor Networks are application-specific, where the design requirements of WSNs change according to the application. Hence, routing protocols requirements are change from application to another. For instant, the requirements of routing protocol that designed for environmental applications is differ from that designed for military or health applications in many aspects. As a result, routing protocols' requirement are as diverse as applications'. Some of these are; Scalability, Latency, Throughput, Recourse Awareness, Data Aggregation, Optimal Route, Over-Head and other metrics. Some applications need some of these metrics to be provided and other applications need the others to be provided. However, routing protocols of all Wireless Sensor networks regardless the application, must or try to maximize the network life time and minimize the energy consumption overall network.

In WSNs the data, based on common phenomena, maybe is collected by many sensors, therefore, there is a probability of data redundancy. Such redundancy is not acceptable to achieve energy and bandwidth utilization. It must be exploited routing protocols. Moreover, Some WSNs are *data-centric* networks sense that data is requested based on attribute-based addressing (composed of a set of attribute-value pair query). For example, query like [temperature > 50°F], means the sensor nodes that sense temperature > 50°F only need to respond and report their readings.

Due to these differences, many new algorithms have been proposed for the routing problem in WSNs taking into account the inherent specification of WSNs along with the application and architecture requirements. In this chapter we have explored some of well-known routing protocols (LEACH, Directed Diffusion, Gossiping and EESR) and their expansions (enhancements), furthermore, their tactics special to WSNs such as data aggregation and in-network processing, clustering, different node role assignment, and data-centric methods.

After that we have compared these explored routing protocols based on different metrics that effect the application requirements in specific and WSN in general. Finally, at the end of this chapter we are going to propose our new protocol and show the simulation result with comparing to its Counterpart.

5.1 LEACH Protocol

Low-Energy Adaptive Clustering Hierarchy (LEACH) is clustering based protocol uses randomized rotation of local cluster base stations. The nodes in LEACH are divided into clusters and each cluster consists of members called Cluster Members and a coordinator node called Cluster Head, CH. The cluster heads are not selected in the static manner that leads to quick die of sensor nodes in the network. However, the randomized protocol has been used in order to balance the energy consumption among the nodes by distributing the CH's role to the other nodes in the network. Furthermore, LEACH uses Time Division Multiple Access (TDMA) protocol in order to regulate the channel access within a cluster (Heinzelman et al, 2000).

It is responsibility of the CHs to assign TDMA slots to the cluster members. The peer to peer communication between the CH and a member is done just during the time slot that assigned to that member, and the other members will be in their sleep state. Hence, it decreases the energy dissipation, see figure 5.1 (Verdone et al, 2007). Moreover, LEACH uses the CDMA communication protocol to decrease the interference between the clusters.

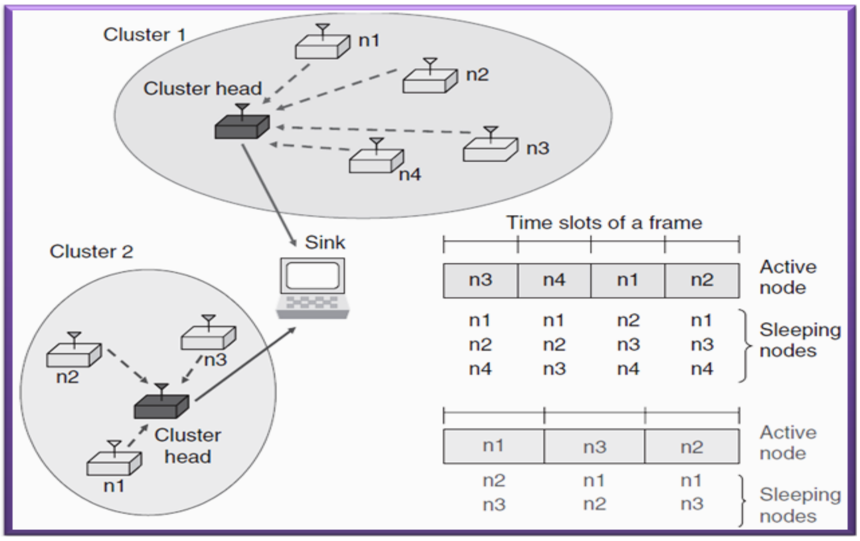

Figure 5.1 LEACH protocol and T DMA schedules

LEACH has been produced to overcome the disadvantages of the Flat-Architecture Protocols that consume more energy (Zheng & Jamalipour, 2009). The CH aggregates/combines the collected data by the nodes to the smaller size and meaningful data and then sends the aggregated data to the sink consuming less energy. LEACH tries to send the data over short distances and reduce number of the transmissions, where the energy consumptions depend on the distance and data size. Therefore, it divides the network into several clusters on rounds each consist of two phases; Setup phase and steady state phase.

Setup phase: During this phase the clusters will be formed and each node decides to be CH or not. The probability of a node, *n*, to be a CH depends on the random number, *P*, that it chooses between 0 and 1. The node *n* become a CH for the current round, *r*, if its probability *P* less than the threshold *T(n)*, where *T(n):*

$$T(i) = \begin{cases} \frac{p}{1-p*(r \bmod \frac{1}{p})} & \text{If } i \in G \\ 0 & \text{Otherwise} \end{cases}$$

Once the CHs are chosen, they will advertise themselves to the other sensor nodes as they are the new CHs by using CSMA protocol. Each node will join the appropriate CH depending on the signal strength received from these cluster heads.

During this phase every non-CH must keep their receiver ON to hear the broadcasts by the CH. Similarly, cluster heads should keep their receivers on to hear these join messages. See the flowchart in the figure 5.2 (Heinzelman, 2000).

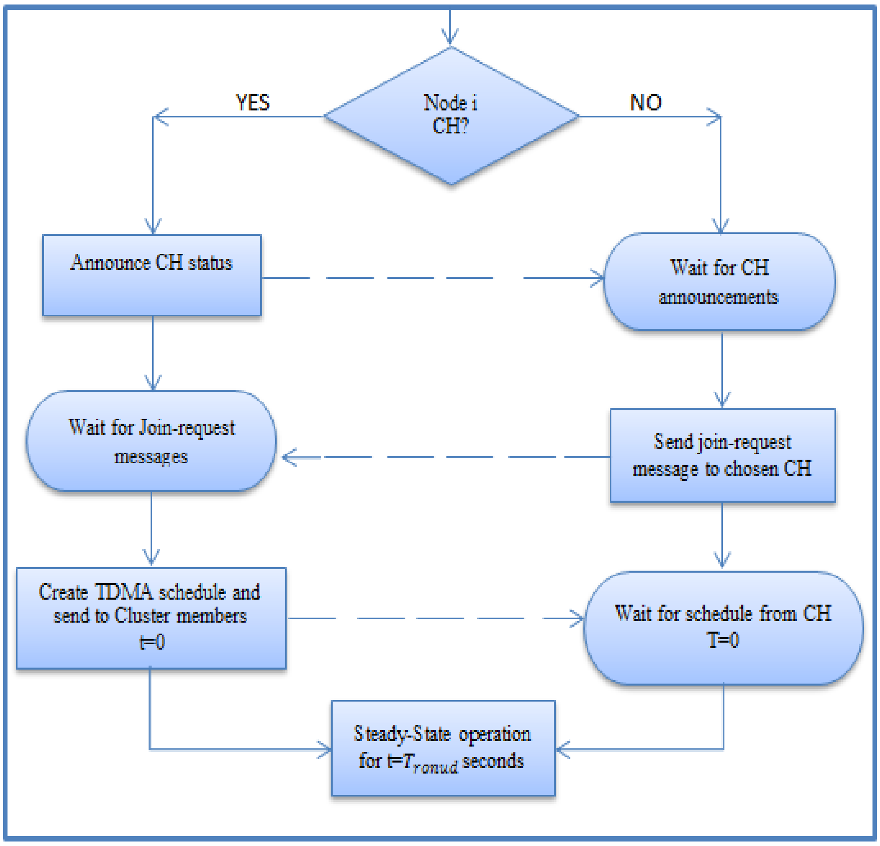

Figure 5.2 Flow-graph of the distributed cluster formation algorithm for LEACH.

Steady state phase: During this phase the data will be sensed and transmitted by the members to their respective CH. Then the CH will aggregate these data and send it to the sink. The network after a certain time will go to another round to the setup phase. As a result, in order to minimize the overhead, the duration of the steady phase is longer than the duration of the setup phase. See the flow graph shown in the figure 5.3 (Heinzelman, 2000).

After a number of rounds, the nodes that have not been a cluster head will have low energy and the CH threshold will be too low. Thus, some CHs will not be able to transmit their data to the base station. Therefore, in order to increase the chance of the node that has not been CH for number of rounds, the threshold equation could be modified by including a factor that increases the threshold of that node (Zheng & Jamalipour, 2009).

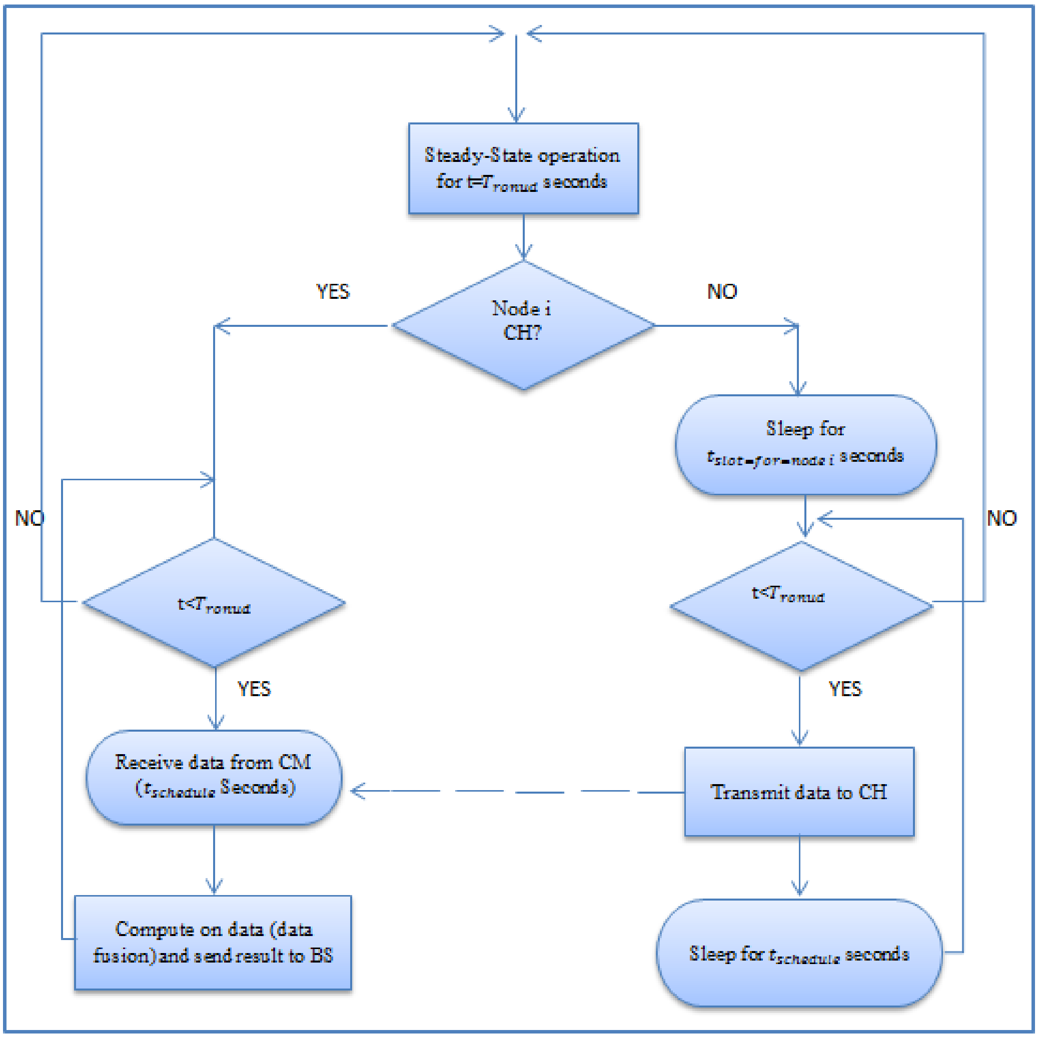

Figure 5.3 Flow-graph of the steady-state operation for LEACH

After the cluster forming is done (see Algorithm 5.1 for the Pseudo-code code that describes the operation of LEACH protocol), the nodes inside the cluster will transmit their data over short distances by dissipating less energy. However, the CHs communicate with the base station directly, hence, consume large amount of energy.

As a result, the main problem with LEACH is the direct send of CH to the sink especially when these CHs are located far away from the sink. However, allowing the multi-hop transmission to the sink through other CHs will solve this issue, where the CH just forwards the data to each other till the sink and does not have to re-aggregate the data come from other CHs.

LEACH, compared to the direct communication and other minimum energy routing protocols, achieves a significant reduction in energy dissipation. Finally, main properties (advantages and disadvantages) of LEACH include (Heinzelman et al, (2000), Zheng & Jamalipour, 2009).

5.1.1 Advantages of LEACH

- It limits most of the communication inside the clusters, hence, provides scalability in the network.
- The CHs aggregates the data collected by the nodes and this leads to limit the traffic generated in the network. Hence, large-scale network without traffic overload could be deployed and better energy efficiency compared to the flat-topology could be achieved.
- Single-hop routing from node to cluster head, hence saving energy.
- Distributiveness, where it distributes the role of CH to the other nodes.
- It increases network lifetime in three ways. First, distributing the role of CH (consume more energy than normal nodes) to the other nodes. Second, aggregating the data by the CHs. Finally, TDMA, which assigned by the CH to its members, puts most of the sensor in sleep mode, especially in event-based applications. Hence, it is able to increase the network lifetime and achieve a more than 7-fold reduction in energy dissipation compared to direct communication (Heinzelman et al, 2000).
- It does not require location information of the nodes to create the clusters. So, it is powerful and simple.
- Finally, it is dynamic clustering and well-suited for applications where constant monitoring is needed and data collection occurs periodically to a centralized location.

ALGORITHM 5.1Pseudo-code describes the operation of the LEACH Protocol

```
Setup Phase:
In this phase clusters are created---cluster heads (CHs) are
chosen

1.       ForEach (node N)
2.         N selects a random number r between 0 and 1
3.         If (r < Threshold value)
3.           N becomes a CH
4.           N broadcasts a message advertising its CH status
5.         Else
6.           N becomes a regular node
7.           N listens to the advertising messages of the CHs
8.           N chooses the CH with the strongest signal as its
               cluster head
9.           N informs the selected CH and becomes a member of
               its cluster
10.        EndIf
11.      ForEach (clusterhead CH)
12.        CH creates a TDMA schedule for each node to transmit
             data
13.        CH communicates the TDMA schedule to each node in the
             cluster
14.      EndFor

Steady State Phase:
1.     ForEach (regular node N)
2.       N collects sensed data
3.       N transmits the sensed data to the CH in the
            corresponding TDMA time slot
4.     EndFor
5.     ForEach (cluster head CH)
6.       CH receives data from the nodes of the cluster
7.       CH aggregates the data
8.       CH transmits the data to the base station
9.     EndFor
```

5.1.2 Disadvantages of LEACH

- It significantly relies on cluster heads and face robustness issues such as failure of the cluster heads.
- Additional overhead due to cluster head changes and calculations leading to energy inefficiency for dynamic clustering in large networks.
- CHs directly communicate with sink, no inter cluster communication, and this need high transmit power. Hence, it does not work well in large-scale networks that need single-hop communication with sink.

- CHs are not uniformly distributed; where CHs could be located at the edges of the cluster.
- CH selection is randomly, that does not take into account energy consumption.
- Finally, it does not work well in the applications that cover large area that requires multi-hop inter cluster communication.

5.1.3 LEACH Enhancements

Due to some drawbacks of LEACH, many researches have been done to make this protocol perform better. Some of these researches are; E-LEACH, TL-LEACH, M-LEACH, LEACH-C and V-LEACH.

E-LEACH: Energy-LEACH protocol improves the CH selection procedure. Like LEACH, it divided into rounds, in the first round all nodes have the same probability to be CH. However, after the first round the remaining energy of each node is different and the node with high residual energy will be chosen as CH rather than those less energy (Xiangning & Yulin, 2007).

TL-LEACH: in LEACH, CH sends the data to the base station in one hop. However, in Two-Level LEACH, CH collects data from the cluster members and relay the data to the base station through a CH that lies between the CH and the base station (Loscri et al, 2005).

***M-LEACH*:** As mentioned above, in LEACH, CH sends the data to the base station in one hop. In Multi-hop-LEACH protocol, the CH sends the data to the sink using the other CHs as relay stations (Zhou, Jiang, & Xiaoyan, 2006). In this protocol the problem of CHs that are away from the base station has been solved, where they were consume huge energy during data transmissions.

***LEACH-C*:** LEACH has no knowledge about the CHs places. However, Centralized LEACH protocol can produce better performance by distributing the cluster heads throughout the network. During the set-up phase, each node sends to the sink its remaining energy and location. The sink then runs a centralized cluster formation algorithm to determine the clusters for that round. However, since this protocol requires location information for all sensors in the network (normally provided by GPS), it is not robust.

***V-LEACH*:** In new Version of LEACH protocol, in addition to having a CH in the cluster, there is a vice-CH that takes the role of the CH when the CH dies (Bani Yassein, Al-zou'bi, Khamayseh, & Mardini, 2009). When a CH die the cluster become useless because the information collected by the node members will not reach the sink.

5.2 Directed Diffusion

Directed diffusion is data-centric routing protocol for collecting and publishing the information in WSNs. It has been developed to address the requirement of data flowing from the sink toward the sensors, i.e., when the sink requests particular information from these sensors (Intanagonwiwat et al, 2000). Its main objective is extending the network life time by realizing essential energy saving. In order to fulfill this objective, it has to keep the interactions among the nodes within a limited environment by message exchanging. Localized interaction that provides multi-path delivery is a unique feature of this protocol. This unique feature with the ability of the nodes to response to the queries of the sink, results in considerable energy savings (Sohraby, Minoli, & Znati, 2007). The main component of Directed Diffusion is *interests*, *data messages*, *gradients*, and *reinforcements*.

The interest (query or interrogation) specifies what the inquirer (user) wants and contains a description of data that interested by the user. Usually, it is expressed using attribute-value pairs as shown in table 5.1(Sohraby et al, 2007).

The data messages are the collected or processed information of a phenomenon or even could be an event that matches the interest or the user’s request.

The gradient indicates the nodes that received the interest. It used to form the reverse paths (multi-paths) towards the sink (inquirer or user).

The reinforcement is one of these paths that constructed in gradient phase which the inquirer (sink) reinforces it by resending the same interest through this specified path.

Table 5.1 Interest Description Using Value and Attribute Pairs (interest in hummingbirds)

Attribute–Value Pair	Description
Type = Hummingbirds	Detect hummingbird location
Interval = 20 ms	Report events every 20 ms
Duration = 10 s	Report for the next 10 s
Field = [(x1,y1),(x2,y2)]	Report from sensors in this area

In order to construct the route between the sink (inquirer) and the sensors that interest to the sink's request, there is four stages; (1) interest propagation, (2) gradient setup, (3) reinforcement, and (4) data delivery. Bellow a detailed description for each stage:

Interest propagation: when a sink want or interest in an event, it initiates the *interest messages* and floods them to all nodes in the network. These messages are exploratory messages indicate the nodes with matching data for the specific task. During this stage, the sink periodically broadcasts the interest message. Once the interest message is received, each sensor node save it in an interest cache, which includes fields for *timestamp* (time of receiving the interest), *gradient* (revers path from the nodes that received the interest towards the sink), *interval*, and *duration* (duration of staying the interest at the cache). After that the nodes flood this message to the other nodes until the node that interest in this interest message, see figure 5.4(a).

Gradient setup: based on local rules different techniques are used to gradient setup. For example, the nodes with highest remaining energy could be chosen when siting up the gradient. During the interest propagation through the network, the gradients from source back to sink will be setup. A node becomes a source node if its observation matches the interest message and sends its data through the gradient path back to the sink as shown in the figure 5.4(b).

Reinforcement: During the gradient setup phase many paths have formed from the source to the sink. Means, the source can send the data to the sink through multiple routes. However, as shown in the figure 5.4(c) the sink reinforces one specific path by resending the same interest through the specified path, which is chosen based on many rules, like, the best link quality, number of packets received from a neighbor, or lowest delay. Along this path each node just forwards the reinforcement to its next hop (Sohraby et al, 2007).

Finally, during this phase, the sink could select multiple paths in order to provide multi-path deliver.

Data Delivery: After the reinforcement phase, as shown in the figure 5.4(d) the route between the source and the sink has been constructed and the data is ready for transmission. For pseudo-code of the directed diffusion algorithm see Algorithm 5.2 (Zheng & Jamalipour, 2009).

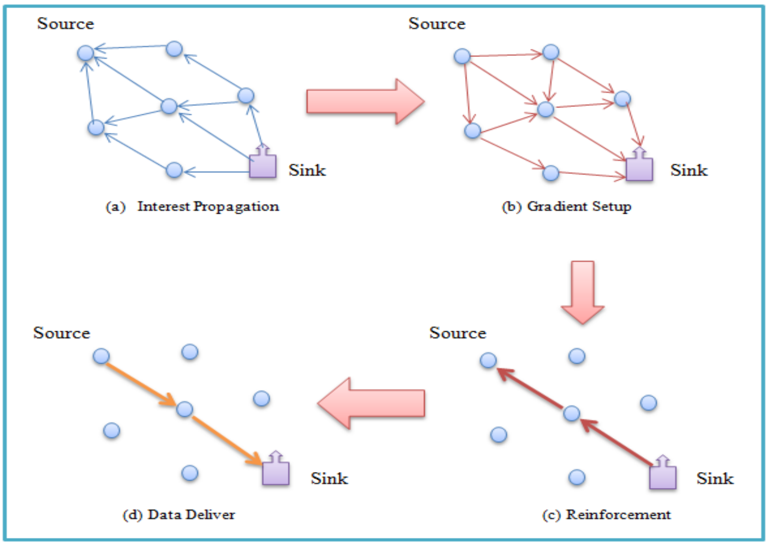

Figure 5.4 Operation of the directed diffusion protocol

Sometimes, for some changes in the WSN, accordingly, the current data route could be dynamically changed. In such case, the sink will send a reinforcement message through new path and a negative reinforcement through the current one as shown in figure 5.5 (drawn from (Akyildiz & Canvuran, 2010)).

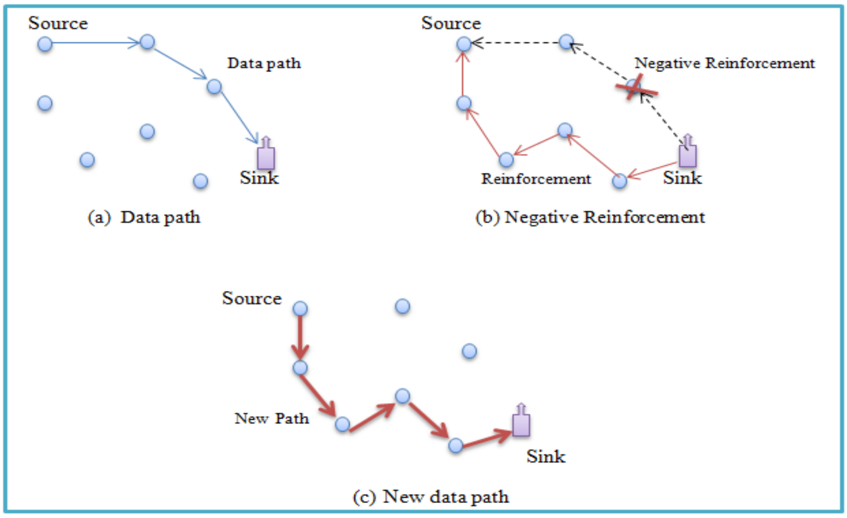

Figure 5.5 Negative reinforcement

ALGORITHM 5.2 Pseudo-code of the direction Diffusion protocol

```
Setup phase:
1.   The base station broadcasts its set of interests
2.   Do
3.    ForEach network node N receiving an interest from node M
4.     N forwards the received interest to its neighbors
         (other than M)
5.     N sets up a gradient with M
6.    EndFor
7.   Until all gradients are set up
8.   Check for loops in the paths and remove them

Operating phase:
1.  ForEach node
2.    Colect sensor data.
3.    Receive messages containing sensor data readings.
4.    Aggregate, correlate or fuse data (if necessary)
5.    If data maches an interest
6.     Forward the data according to the gradient associated
        with the interest
7.    EndIf
8.  EndFor
```

As a result, we can say the Directed Diffusion is characterized by these following specifications (Advantages and disadvantages) (Akyildiz & Canvuran, (2010), Eroglu, (2006).

5.2.1 Advantages of Directed Diffusion

- It is designed to retrieve data aggregates from single node.
- Data is named by attributed-value pairs (table 5.1)
- It works well in multipurpose wireless sensor network and in sensor networks that query, for example, 'GIVE ME THE TEMPERATURE IN PARTICULAR AREA' or 'WHO SEE THE BLACK COW'.
- Not like other routing algorithms, in Directed Diffusion at the same time more than one sink can make queries and receive data, hence, simultaneous queries could be handled inside a single network.

- The interests/queries are issued by the sink not by the sources and just when there is a request (demand-based). Moreover, all communication is neighbor-to-neighbor (like GOSSIPING), which removes the need for addressing and permits each node to aggregate data. As a result, both points contribute to reduce energy consumption.
- It provides application-dependent routes based on the interests of the user.
- In order to satisfy the user's requests, network routes are changed according to sensor readings changes.
- It does not require nor global node addressing mechanism neither a global network topology. Moreover, the routes are formed just when there is an interest. As a result, it achieves energy efficiency.
- It mostly selects a specific route for the interest. Hence, it decreases the energy consumption in the network.
- The nodes that have matching information are only the nodes that involved in the information generation.

5.2.2 Disadvantages of Directed Diffusion

- It is, generally, based on a flat topology. Hence, scalability and congestion (especially in the nodes that near to the sink) problems are exist.
- An overhead problem occurs at the sensors during the matching process for data and queries.
- In Directed Diffusion, the initial interest contains a low data rate. However, an important overhead caused during flooding operation of interest propagation phase.
- Due to the flooding required to propagate the interest en establish gradients on each node, it is not optimized for energy efficiency and need high amounts of memory to store interest gradients and received messages.
- It mostly selects the shortest path between the source and the destination, which leads to quickly death of nodes on that path (Li, Ma, Wang, & Tan, 2011).
- Finally, Directed Diffusion is a query-based protocol. It may be not work well in application that continuous data transfers are required (dynamic applications), for instant, environmental monitoring applications.

5.3 EESR Protocol

Energy-Efficient Sensor Routing (EESR) is a flat routing algorithm (Oh & Chae, 2007) proposed especially to reduce the energy consumption and data latency, and to provide scalability in the WSN. Mainly, it consists of *Gateway*, *Base Station*, *Manager Nodes*, and *Sensor Nodes* (Oh, Bahn, & Chae, 2005). Their duties are: ***Gateway:*** Deliver messages from Manager Nodes or form other networks to the Base Station. ***Base Station:*** Has extra specification than normal sensor nodes. It sends and receives messages to/from Gateway. Moreover, it sends queries and collect data to/from sensor nodes. ***Manager Nodes and Sensor Nodes:*** collect data from the environment and send it to each other in 1-Hop distance till the Base Station.

Application area is divided based on the 2-dimensional (x, y) coordinates into four quadrants; (+ +), (+ -), (- -), and (- +), and the Base Station is located in the center (at coordinate). Furthermore, each quadrant, in turn, is divided into sectors, locating the Base Station in the middle, their numbers determined by minimum hops required to deliver a message from the base station to the farthest position in the quadrant. Manager Nodes are located (predetermined) in the center of each sector on the diagonal line of the quadrant with 1-hop distance between each other. Finally, the other nodes are randomly distributed in the application area, see figure 5.6(taken from (Oh & Chae, 2007)).

As shown in the figure 5.6, each quadrant has three sectors because the Base Station can communicate with the furthest node minimum in 3-hops. Each sector has its own ID, gated it from Base Station, determined by the quadrant name and the distance from the base station. For example, 1-hop distance sectors names are (+1+1) sector, (+1-1) sector, (-1-1) sector, and (-1+1) sector.

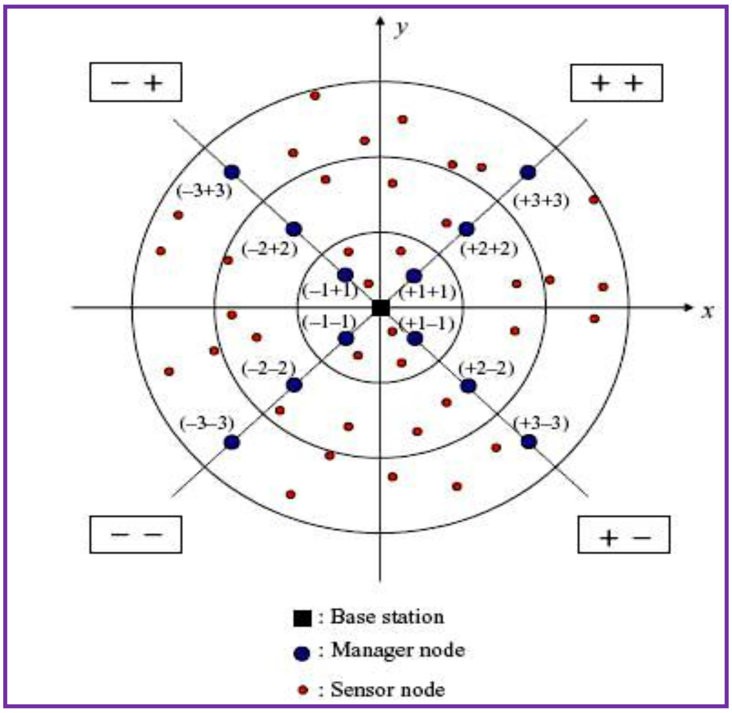

Figure 5.6 Locations of the nodes based on 2-dimensional (x, y) Coordinates

Each sensor node constructs its EESR table, as shown in the table 5.2, by broadcasting a “HELLO” message within 1-hop neighbor. The table contains distance from the base station, Quadrant Names, Sector ID and Manager Node Names.

Table 5.2 Quadrant Names, Sector ID, and Manager Node Names (according to figure 5.6)

Distance from the base station	Quadrant name	Sector ID	Manager Node name
1 hop	(+ +) (+ -) (- -) (- +)	(+1+1)sector (+1 -1)sector (-1 -1)sector (-1 +1)sector	+1 +1M.N +1 -1M.N -1 -1M.N -1 +1M.N
2 hop	(+ +) (+ -) (- -) (- +)	(+2 +2)sector (+2 -2)sector (-2 -2)sector (-2 +2)sector	+2 +2M.N +2 -2M.N -2 -2M.N -2 +2M.N
3 hop	(+ +) (+ -) (- -) (- +)	(+3 +3)sector (+3 -3)sector (-3 -3)sector (-3 +3)sector	+3 +3M.N +3 -3M.N -3 -3M.N -3 +3M.N

Work of the algorithm

After the nodes are deployed, the Base Station sends the relative direction information and sector ID of each node, then each node construct its EESR table. Once a node detects an event, in order to select the next node to deliver the event, it investigates the sector ID of all neighbor nodes within 1-hop in its EESR table. The node selects its next node in one of these three procedures:

- If a Manager node is within 1-hop distance, it will be the next hop.
- If there is no Manager node, it will check for a normal 1-hop distance node that exists on the same sector to be the next hop.
- Otherwise it will look to another node that lies out of its sector but close to the Base Station to be the next hop. The nodes that lie on the same quadrant are preferred ones

After selecting next neighbor node, the first node will send the event only to this selected node. Once the selected node receives the event it, in turn, repeats the same procedure to select its next 1-hop and send the event. This process will continue till the Base Station receives the event. However, if a Manager Node receives the event, the event will transmit from manager-manager until the Base Station, as shown in the flow chart in the figure 5.7 (redrawn from (Oh & Chae, 2007)).

After receiving the first event, the next hop (next neighbor node) will be selected easily for those nodes that sent data before by storing the EESR table of each node to result of selection procedure of the next node (three procedures have mentioned above) according to the sector ID.

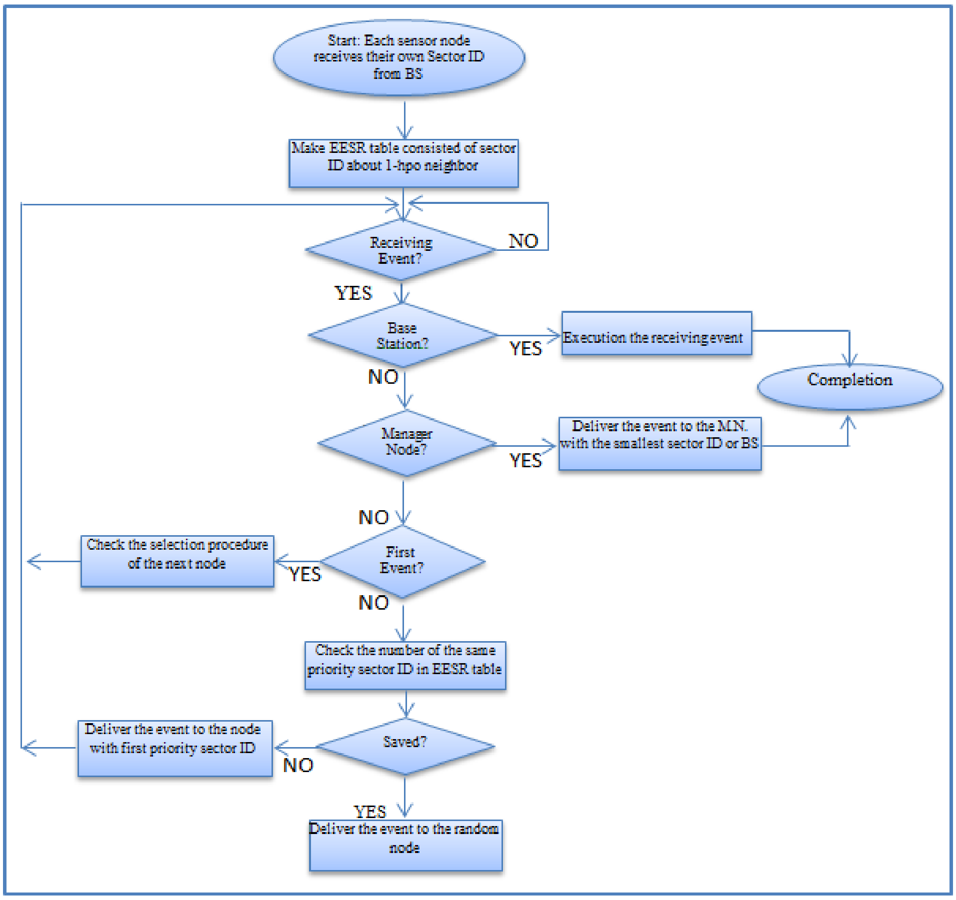

Figure 5.7 the procedure of the EESR scheme

5.3.1 Advantages of EESR

- It divides the application area into Sectors, hence, it is scalable.
- It energy-efficient and achieves this feature in three ways; first, it sends the event to the just one node and does not flood it, second, Manager Nodes rely the data in

predefined shortest path, and finally, normal nodes after sending the first event they will easily select the next node by using their EESR tables. As a result, it consumes little energy and prolongs the network life time.

- It is One-one communication. Moreover, after sending the first event, the next hop will be found easily. As a result, it is low latency.
- In order not to send the data through a same route and exhaust energy of these nodes, sometimes, it chooses other routes to deliver the data.

5.3.2 Disadvantages of EESR

- All 1-hop nodes of the event detected node could be out of it transmission range. So, it has no specific criterion to select the next node.
- If a node located in the furthest sector detects an event and the next hop is located in the lower sector, the data will be lost in case of the lower node's energy has had finished.
- If the normal nodes that located in the furthest sector detect an event and accidently every times their next hop is Manager Nodes, the energy of these Manager Nodes will exhaust earlier. Because they will send the event manager-manager until the Base Station.
- There is no balance in energy consumption, where some nodes consume their energy before other nodes.

5.4 Gossiping Protocol

Gossiping (Hedetniemi, Hedetniemi & Liestman, 1988) is data-relay protocol, like Flooding protocol, does not need routing tables and topology maintenance. It was produced as an enhancement for Flooding and to overcome the drawbacks of Flooding, i.e., implosion. In Flooding, a node broadcasts the data to the all of its neighbors even if the received node has just received the same data from another node. The broadcasting will continue until the data will be received by the destination. However, in Gossiping, a node randomly chooses one of its neighbors to forward the packet to, once the selected neighbor node receives the packet it chooses, in turn, another random neighbor and forwards the packet to. This process will continue till the destination or number of hop has been exceeded. As a result, just the selected nodes/neighbors will forward the received packet to the sink. Unlike Flooding, Gossiping is serves well at one-to-one communication scenarios but it does not at one-to-many. Packet forwarding mechanism for both Flooding and Gossiping are shown in figure 5.8 (Verdone et al, 2007).

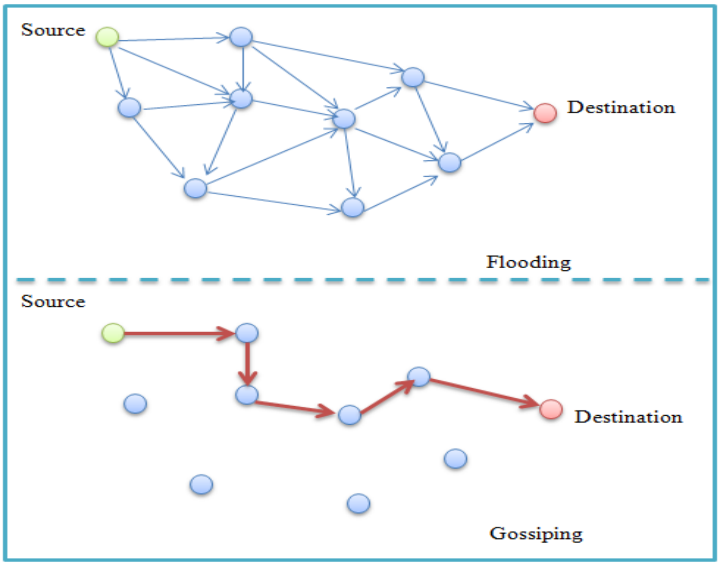

Figure 5.8 forwarding mechanism of both Flooding and Gossiping

Main objective of Gossiping was reducing the power consumption and keeping the routing system as simple as possible. However, it suffer from the latency that caused by the data propagation. The consumed power by Gossiping (Zhang & Cheng, 2004), is approximately equal to

$$O(K^{L})$$

K: number of nodes that forward the packet.

L: number of hops before the forwarding stops.

The most considerable feature of Gossiping is the ability of controlling the power consumption by selecting appropriate K and L.

5.4.1 Advantages of Gossiping

- It is very simple does not need any routing table and topology maintenance. So, it consumes little energy.
- It is appeared as an enhancement to overcome the implosion that exists in Flooding.
- In Gossiping just the selected nodes contribute in forwarding the data to the sink.
- It works well in applications that need one-to-one communication but it does not in one-to-many.

5.4.2 Disadvantages of Gossiping

- The next hop neighbor is randomly chosen, this means, it may include the source itself.
- The packet will travel through these selected neighbor until it reaches the sink or number of hops exceeds
- It is suffer from packet loss.
- The remarkable disadvantage of Gossiping is suffering from latency that caused by data propagation.

5.4.3 Gossiping Enhancements

In order to enhance the Gossiping protocol, many protocols have been produced as an expending to it. For example, FLOSSIPIN, SGDF, LGossiping and ELGossiping,

FLOSSIPING Protocol (Zhang & Cheng, 2004)***:*** It combines the approaches of both flooding and the gossiping routing protocols. When a node has a packet to send, it decide a threshold and save it in the packet header then randomly selects a neighbor to send the packet in Gossiping mode, while the other neighbor nodes listen to this packet and generate a random number. The neighbors that their generated random numbers are smaller than the threshold will broadcast the packet in Flooding mode. As a result, the Flossiping improves the packet overhead in Flooding and the delay issue in the Gossiping.

- ***SGDF Protocol*** (Yen, Chen, & Yang, 2008)***:*** **S**ingle **G**ossiping with **D**irectional **F**looding routing protocol divided into two phases; Network Topology Initialization and Routing Scheme. In the first phase, each node generates a *gradient* (shows number of hops to the sink). In the second phase, in order to deliver the packet, SGDF uses single gossiping and directional flooding routing schemes. As a result, SGDF achieves high packet delivery ratio, low message complexity, and short packet delay.

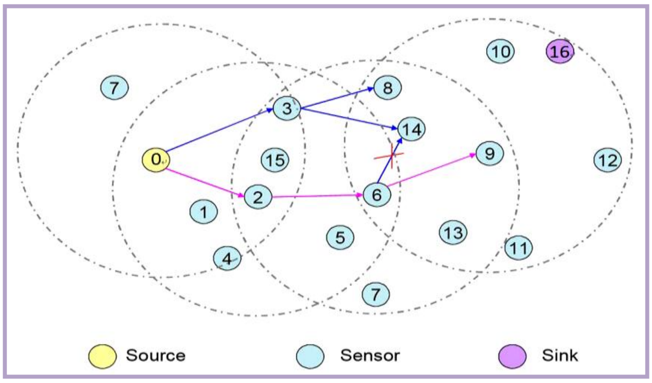

Figure 5.9 Routing Scenario in SGDF

- ***LGossiping Protocol*** (Kheiri, Ghaznavi Goushchi, Rafiee ,& Seyfe, 2009)***:*** In **L**ocation based **Gossiping** protocol, when a node has an event to send, it randomly chooses a neighbor node in its transmission radius. Once the neighbor node receives this event, it, in turn, randomly chooses another node within its transmission radius and sends it. This process will continue until the sink. As a result, delay problem has been solved to some extent. See figure 5.10.

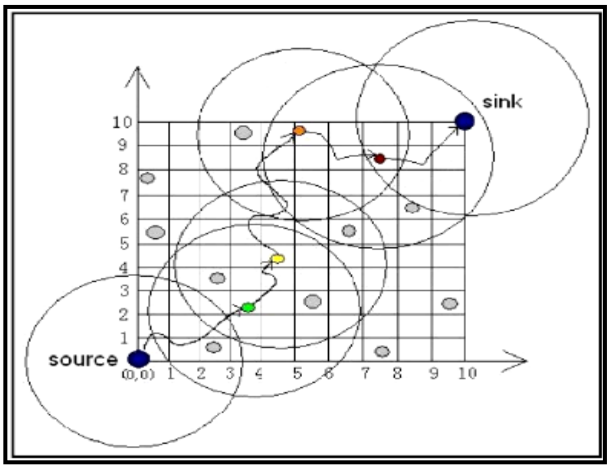

Figure 5.10 Schematic of Data Routing in LGossiping

5.5 Comparison of Explored Routing Protocols

During this research, many differences have been observed, generally, between flat and hierarchical routing protocols and, precisely, among these researched routing protocols. When compared to the other protocols, Gossiping, is very simple does not need any routing table or topology management. It provides very high connectivity, where, as soon as a node becomes aware of its neighbors it is able to send and forward packets. Gossiping protocol is based on the flooding protocol. Instead of broadcasting a packet to all neighbors, the packet is sent randomly to a single neighbor, means, only one copy of a packet is in transit at any one time. Having received the packet, the neighbor chooses another random node to send the packet to. However, this may include the node which sent the packet itself. This process continues until the packet reaches its destination or the maximum hop count of the packet is exceeded. As a result, compared to LEACH, Directed Diffusion and EESR Protocols, Gossiping uses a medium amount of power.

Gossiping, compared to other Protocols, suffers from quite **high** latency because of the data propagation through network (one to one communication) and the hop count could become quite large due to random nature of the protocol. As the number of nodes in a network increases, the number of paths that a packet can follow increases. On average the number of hops taken to traverse the network increase. Hence, packets are dropped when the packets hop count reaches a maximum value. In larger networks it is more likely that a packets hop count will reach this value and so more packets are dropped. In smaller networks roughly half of the packets sent are lost and in larger networks the loss rate increases drastically. As a result, Gossiping protocol is the worst protocol in terms of packet lost. Hence, Gossiping is not Scalable like LEACH, Directed Diffusion and EESR.

As a result, we summarized all what mentioned above in two tables; table 5.3, shows a general comparison of different routing approaches for flat and hierarchical sensor networks (Al_karaki & Kamal, 2004) and table 5.4 shows how these researched routing protocols (LEACH, Directed Diffusion, EESR and Gossiping) fit under different categories and also compares these routing techniques according to many metrics.

Table 5.3 General Comparison between Flat and Hierarchical Routing Protocols

Hierarchical Routing	Flat Routing
Reservation-based scheduling	Contention-based scheduling
Collisions avoided	Collision overhead present
Reduced duty cycle due to periodic sleeping	Variable duty cycle by controlling sleep time of nodes
Data aggregation by cluster-head	Node on multi-hop path aggregates incoming data from neighbors
Simple but non-optimal routing	Routing can be made optimal but with an added complexity.
Requires global and local synchronization	Links formed on the fly without synchronization
Overhead of cluster formation throughout the network	Routes formed only in regions that have data for transmission
Lower latency as multiple hops network formed by cluster- heads always available	Latency in waking up intermediate nodes and setting up the multipath
Energy dissipation is uniform	Energy dissipation depends on traffic patterns
Energy dissipation cannot be controlled	Energy dissipation adapts to traffic pattern
Fair channel allocation	Fairness not guaranteed

Table 5.4 Comparison between LEACH, Directed Diffusion and Gossiping Routing Protocols

	LEACH	Directed Diffusion	EESR	Gossiping
Class	Hierarchical	Flat	Flat	Flat
Scalability	High (Divides the nodes into clusters)	Limited (Due to flat topology nature)	High (Divides application area to sectors)	Limited (Due to flat topology nature)
Life Time	Very Good (Due to TDMA, most the sensors are in sleep mode and distributing the role of CH to other nodes)	Good (demand-based and neighbor-neighbor Communication)	Very Good (Due to predetermined allocation of M.N. And using sectors)	Medium (It suffer from high latency)
Energy efficient	HIGH (Single-hop routing from node to cluster head)	HIGH (mostly selects a specific route for the interest and routes are formed just when there is an interest)	HIGH (It sends only to 1-hop neighbors and chooses different routes to data deliver)	Medium (Not like LEACH or EESR. However, no routing table and topology maintenance)
Data aggregation	YES (data aggregated by CHs)	YES (each node aggregate data then relay it to the next hop)	Yes	NO (The node that participate in data delivery are just relay stations)
Negotiation-based	NO According to signal strength	YES (Negotiation is done during Gradient setup phase)	YES (constructing EESR table finding the optimum route)	NO (A node randomly select a neighbor node to send)
Recourse Awareness	YES	YES	NO	NO
Hop Comm.	Single-Hop Member-CH and CH-BS	Multi- Hop (From source to BS through other nodes)	Multi-Hop (Each time 1-hop till B.S.)	Multi-Hop (From source to BS through other nodes)
Optimal Route	NO Member has one chance rely to CH	YES (Reinforcement phase)	YES (Every time checks its EESR table to find better Sector ID)	NO (Randomly select)
Latency	Little Node directly sends its data to CH	High (due to the flooding during Interest propagation)	Little (No over-head, and After the first event, the next hop will be found easily)	Very High (Due to data propagation)
Throughput	Very high No delay and only one node access the channel at a time	Acceptable	High (Select optimum path, no delay, no over-head)	Low (Due to the high delay)
Over-head	NO CH aggregates the data of many nodes	YES (Overhead during the matching process for data and queries)	NO (One to one communication)	NO (it sends to a neighbor node directly)
-based	Event-based Just when an event occur the sensor detect it	Query-based (The queries are issued by user just when there is a request)	Event-based (Only when an event occur the sensor detect it)	Event-based (only when an event occur the sensor detect it)
Applications	Monitoring app. (Dynamic app.) If an event occur a node detect it	Multi-purposes Applications (at the same time more than one sink can make queries and receive data)	Monitoring app. (Dynamic app.)	Application need (one-to-one communication)
App. Type	Health monitoring (artificial Retina)	Environmental monitoring (PODS Hawaii)	environmental monitoring i.e., Agricultural application	Environmental monitoring or during deployment phase and network initialization

As mentioned above and seen from table 5.4, although the simplicity of Gossiping, it has many disadvantages compered to LEACH, Directed Diffusion and EESR. Accordingly, many protocols have been proposed to overcome these drawbacks of Gossiping Protocol. However, none of them has addressed problems of Gossiping completely. For example, FLOSSIPN, which combines Flooding and Gossiping, it solved the overhead of flooding and the delay of Gossiping but the power consumption and packet delay time in this protocol are the same of the flooding and the gossiping routing protocols. SGDF achieves high packet delivery ratio, low message complexity, and short packet delay. However, because of the direct flooding, amount of packets becomes larger during packet delivery. LGossiping has chosen the delay problem to some extent, but problem of non-reaching packets to the Base Station still exists. Moreover, this protocol (LGossiping) uses GPS to determine location of each node. Hence, additional hardware means extra money.

As a result, none of these solutions has completely solved the Gossiping problems. Therefore, in the next chapter, we are going to propose a new protocol as improvement to LGossiping. In this new proposed protocol we will try to reduce the delay caused by packet propagation, solve the problem of non-reaching packets and increase the network life time with keeping in mind way of working the LGossiping.

CHAPTER SIX
PROPOSED PROTOCOL (TWELGossiping)

6.1 TWELGossiping Protocol

In this section we are going to propose our new protocol, the **T**wo **W**ays **E**fficient **L**ocation-based **G**ossiping protocol (TWELGossiping), to address the problems of Gossiping and its extensions. TWELGossiping consists of two phases; Initialization Phase and Routing Phase. In the first phase each node generates the gradient to the sink/base station. While in the second phase, after the gradient has constructed, the TWELGossiping uses two ways (two routes) to deliver the packets to the base station. Then we are going to show how our proposed protocol reduces the propagation delay and non-reaching packets to the base station. Moreover, this protocol will maximize the network life time with compared to its Counterpart. The two phases are explained in detail in the following.

6.1.1 Network Initialization Phase

The network initialization phase starts after the sensor nodes are randomly distributed in the application area. In the beginning, the base station broadcasts a "HELLO" message to its neighbors. The HELLO message contains: Base Station Address fixed and Hop count (initial number = 1) variable. The Hop count used to setup the gradient to the Base Station. Means, it shows the node distance/depth to the Base Station. After broadcasting the HELLO massage all 1-hop neighbors of the Base Station will receive this message and get the Base Station address and the hop count. Each node saves the hop count in its memory and increases the hop count by 1. Then the new hop count is replaced with old one. After that each node that received the HELLO message will continue to broadcast this message to the farther nodes. As shown in the figure 6.1 in each stage the hop count will be incremented by 1.

BS Address	Hop Count.

HELLO message format

When a node receives a HELLO message, it will check if it already has a gradient. If it has a gradient, it will compare the hop count of the message with its own, and it will replace its hop count with the message's one if the message's hop count is smaller than its hop count and add 1 to the hop count and broadcast it. However, if its hop count smaller than or equal to the hop count of the message, it will discard the message. This case occurs because of the message broadcasting, where the message reaches the node through different routes. As a result, the gradient will keep the best route. Finally, the process will continue until all the sensor receive the HELLO message, at that time the network initialization phase will be completed. Now each node through the gradient knows its distance to the Base Station. Figure 6.2 summarizes the setup phase in a flowchart.

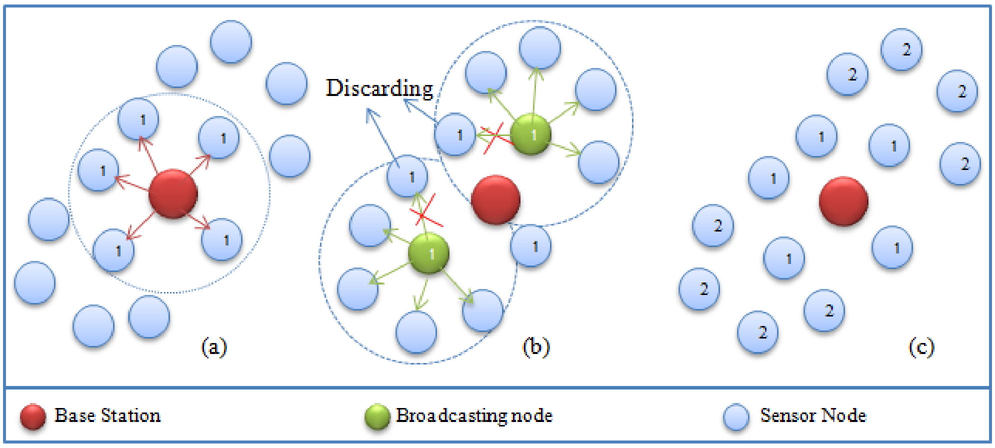

Figure 6.1 TWELGossiping Network Initialization phase

Figure 6.1 shows how the HELLO message is broadcasted through the network. In figure 6.1(a) 1-hop neighbor for the Base Station will store the hop count in their memory and start constructing the gradient. Each node increment the message hop count by 1 and broadcast the message with the new hop count (at this time becomes 2). In figure 6.1(b), we can see that the HELLO message has reached 1-hop neighbors; two of them become broadcasting nodes which in turn continue broadcasting the message. Then the neighbors of the broadcasting nodes receive the message and compare the hop count of the message with their own. If their own hop count is equal or smaller than the message's hop count they will discard the message as shown in the figure 6.1(b). The process will progress until completing the initialization phase as shown in the figure 6.1(c).

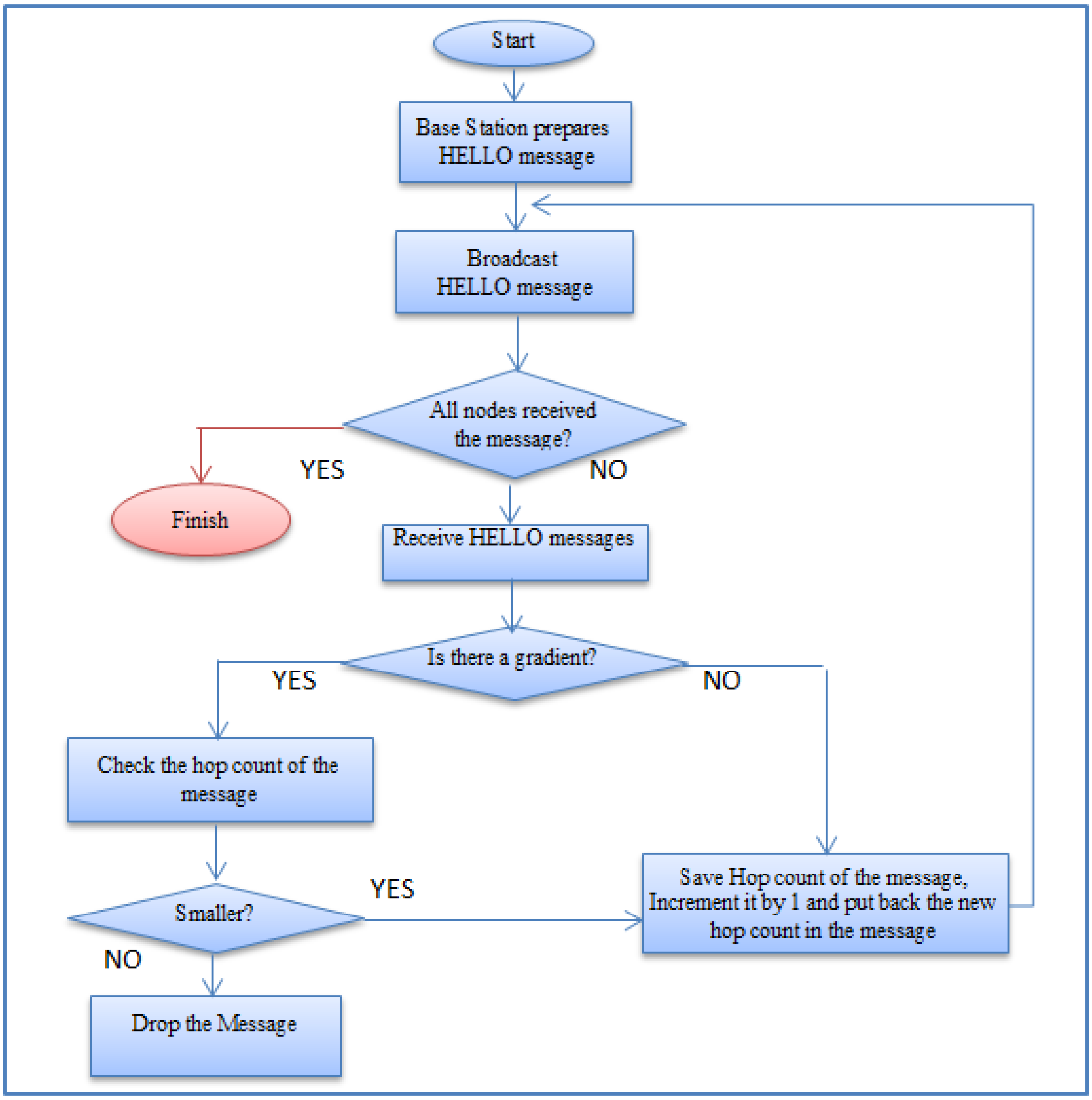

Figure 6.2 flowchart of TWELGossiping Network Initialization phase

6.1.2 Routing Phase

After the network initialization phase finishes, the routing phase will start. This phase is consists of two stages; Event Detected Node Stage and Two ways stage and these two stages, as shown in the flowchart (figure 6.4), have been split by the dotted line. Here we have some assumptions:

- At the beginning all the nodes are full of energy and have the same amount of energy.
- Each node knows its remaining energy at any time.
- Each node has a transmission radius of 30 m.

Event Detected Node Stage: When a node detects an event and wants to send it to the Base Station, it generates a packet its header consists of six fields as shown below:

Source	***Destination***	***Present***	***Next***	***Hop***	***TTL***

Header Packet

Source: Source address of the source node (event detected node).
Destination: Address of the Base Station.
Present: Address of the current node.
Next: Address of the next hop.
Hop: Hop count of the next node to the Base Station (number).
TTL: Time To Live of the packet equal to (2 × hop count of the source). The packet will be dropped if this field becomes 0.

Then within its transmission radius it broadcasts a request message to its neighbors asking them for their residual energy and their hop count to the Base Station. The received neighbors will reply to this request by sending their residual energy and hop count. Accordingly, the source chooses two neighbors that have the minimum hop count (smaller than its hop count) and maximum residual energy to be the next hop and sends the packet to these two neighbors. however If there are no neighbors with smaller hop count than the source, the source will choose neighbors with hop count equal to its hop count otherwise will select randomly any two nodes to be the next hop of the packet.

Once these two neighbors receive the packet, (Event Detected Node Stages will finishes and Two Ways Stage will start), each one regardless the other, will repeat the same process described above. However, this time just choosing **one neighbor** to be the next hop for the packet. Means, the packet will travel through two routes (two ways) towards the Base Station as shown in the figure 6.3. An important point to mention about is that, energy reduction for each node occurs at every transmission or reception has been made. Hence, probability of choosing the same node to be next hop will reduce. Thereby, the energy has fairly used.

As soon as the next hop is selected its address will be written in the **next field** of the header. The source will write its address in the **source field** and **present field** just for the node that detected the event. After that the **source field** (address of the event detected node) will be fixed but the **present field** will change according to the present node. In the **Hop field** of the header the hop count of the next node will be written. Finally, **(**2 × hop count of the source) will be put in the **TTL field,** taking into account that the source will not find a neighbor with hop count smaller than its hop count, and in this case it selects a neighbor with hop count equal to its own. Each node receives the packet will subtract 1 from TTL field and progress to traverse the packet towards the Base Station. The packet relaying will continue until the Base Station receives the packet or the TTL field becomes 0.

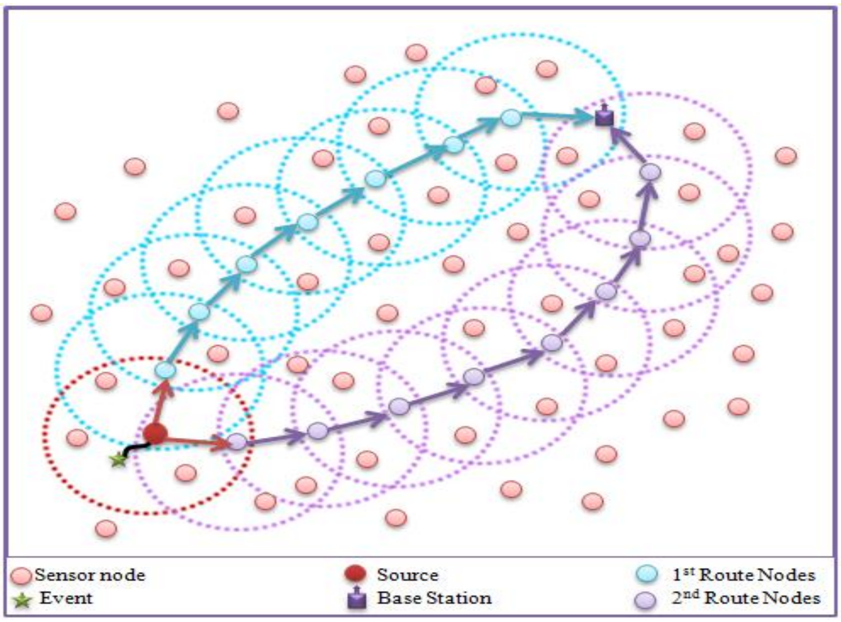

Figure 6.3 Routing Phase in TWELGossiping

When the Base Station receives the packet, it will broadcast a declaration message of successful receiving to inform the other nodes that try to send the same packet through other route to drop the packet. This declaration message prevents the same message from reaching the Base Station two times, reduces the messages overhead and the energy consumed by the other nodes during packet travelling through other route. Figure 6.4 briefs the Routing Phase in a simple flowchart and Algorithm 6.1 shows the Pseudo-code both of Initialization and Routing Phases.

As a result, in our proposed protocol, firstly; we have extended the network life time through fair using of the energy by selecting a node with a maximum residual energy and lower distance to the Base Station. Secondly, we have achieved high packet delivery ratio (number of non-reaching nodes have been reduced) and reduce the delay of delivering the packet by sending the same packet through two routes. Thirdly, we have reduced the message overheads and the energy consumed by the nodes that already trying to send the data to the Base Station by sending the acknowledgement message of successful receiving the packet.

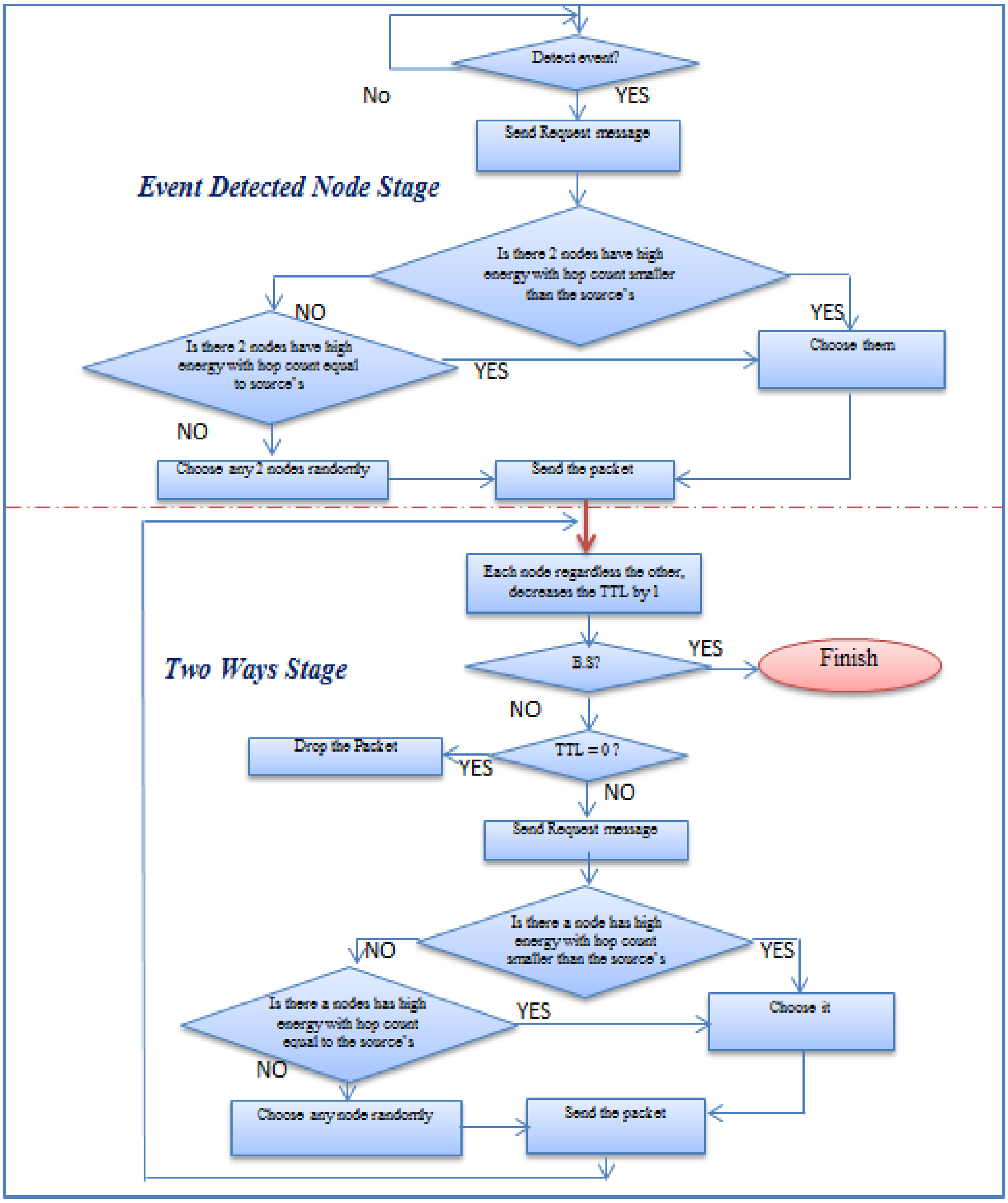

Figure 6.4 flow chart of Routing Phase in TWELGossiping

Algorithm 6.1 Pseudo-code both of Initialization and Routing Phases

Initialization Phase

1. Base Station broadcasts the HELLO message
2. DO
3. FOR each node receiving the HELLO message
4. DO
5. Checks for gradient if it is already exist
6. Saves the least hop count (gradient) into its memory
7. Increases the hop count of the message by 1 and continues broadcasting
   the HELLO Message to other neighbors
8. END
9. Repeat until all nodes have their hop count (gradient) to the Base Station
10. END FOR
11. END

Routing Phase

1. Event detected node within its transmission radius broadcasts a request message
2. DO
3. Chooses 2 nodes with high residual energy have hop count **less** than its Own
4. Otherwise
5. Chooses 2 nodes with high residual energy have hop count **equal** to its Own
6. Otherwise
7. Chooses any 2 nodes randomly
8. Send DATA
9. FOR each node receiving the DATA regardless the other decreases 1 from
   TTL field and within its transmission radius broadcasts a request message
10. Chooses a node with high residual energy has hop count **less** than its Own
11. Otherwise
12. Chooses a node with high residual energy has hop count **equal** to its Own
13. Otherwise
14. Chooses any nodes randomly
15. Forward the DATA
16. Repeat until the DATA reaches the Base Station or TTL field becomes 0
17. END FOR
18. END

CHAPTER SEVEN
SIMULATION SETUP AND EXPERIMENTAL RESULTS

7.1 Simulation Setup

First of all, after understanding that the wireless communication component of a sensor node is responsible for the energy draining activities, we have used the same radio model of (Heinzelman, Chandrakasan, Balakrishnan, 2000). The model is shown in the figure 7.1. The first order radio model offers an evaluation of energy consumed when transmission or reception is made by a sensor node at each cycle. The radio has a power control to expend minimum energy required to reach the intended recipients.

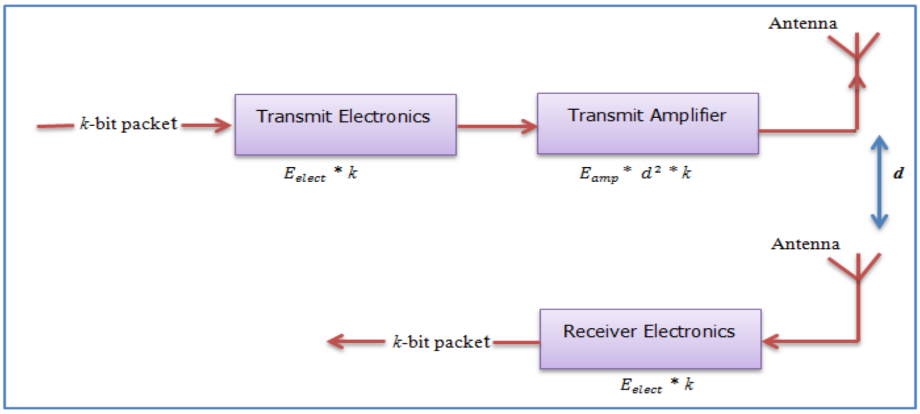

Figure 7.1 Schematic diagram of the first order radio model equation

Mathematically, when a *k*-bit message is transmitted through a distance, *d*, required energy can be expressed as stated in the equation:

$$E_{Tx} = (E_{elect} \times k) + (E_{amp} \times d^2 \times k) \qquad (7.1)$$

Likewise, the energy consumed at the reception is illustrated as shown in the equation:

$$E_{Rx} = E_{elect} \times k \qquad (7.2)$$

Where
$E_{Tx-elect}$: is energy dissipated per bit at transmitter

$E_{Rx-elect}$: is energy dissipated per bit at receiver
E_{elect} : is cost of circuit energy when transmitting or receiving one bit of data
E_{amp} : is amplifier coefficient
k : is a number of transmitted data bits
d : is distance between a sensor node and other or between a sensor node and the base station.

In order to simulate the behavior of Wireless Sensor Network, there are several simulation tools available such as NS-2, OMENT++, MATLAB, OPNET and others. In this thesis the proposed protocol is simulated using Network Simulator 2 (NS-2), using perl programming to analyze the results of the trace file in terms of average end-to-end delay, energy consumption or energy saving over all network, average number of hops, and average number of packet lost which is the main issue of gossiping, And finally, using MATLAB as drawn tool just to draw the plots of these metrics by entering the Perl programing results.

NS-2 is a discrete event and object-oriented network simulator. It is used in most ad-hoc networking research to simulate routing, multicast protocol and a variety of network components. In NS-2 a trace file is generated during simulation time which contains all events in the network, e.g., links, nodes, as well as packet traces. The associated Network Animator (NAM) is an animation tool for viewing the events that happened in the network depending on the trace file information. Our work is more flexible and sufficient modifications can be done using NS-2.

In this simulation, a total number of 100 nodes are randomly deployed on an area of 100m x 100m. Figure 7.2 illustrates the simulated environment (topology) of the 100 nodes we deployed.

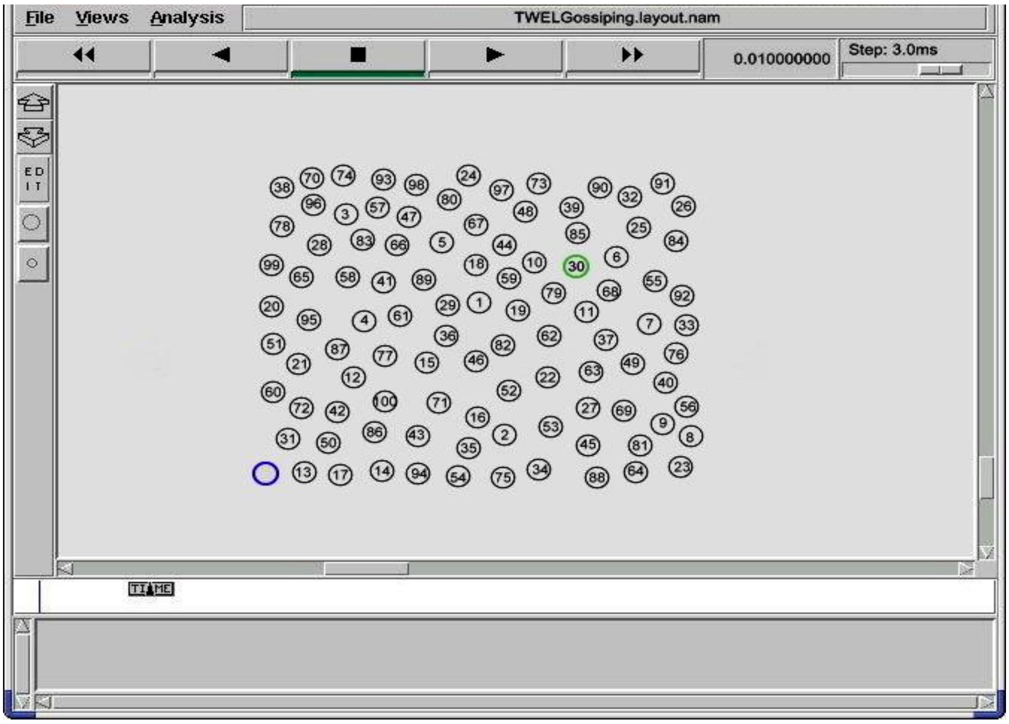

Figure 7.2 Deployment of 100 nodes randomly in an area of $100m \times 100m$

With the nodes being deployed, some assumptions were made concerning the node features and these are as follows:

- All nodes are homogeneous in nature.
- Nodes are static.
- All nodes start with the same initial energy.
- The base station is placed at the (0,0) origin of the area space (blue circle in figure 7.2)
- At every transmission or reception made, energy reduction occurs for every node. Hence, probability of choosing the same node to be next hop will reduce.

The parameters used in the simulation are listed in Table 7.2:

Table 7.1 The initial parameter of our simulation

Parameter	Quantity
Total number of nodes, (N)	100
Initial energy of each node (Joules), (E_{in}(n))	100
Packet size (k) in bytes	100
Energy circuitry cost at transmission and reception of a bit of data (E_{elect}) in nano Joule per byte	75
Amplifier coefficient (E_{amp}) in pico Joule per bit	100
Coordinate of base station	(0,0)
Transmission Range of each node (in meter)	20

7.2 Performance Parameters

7.2.1 Number of Lost Packet

Being packet lost the main issue in Gossiping protocol, it is the most important metric used to compare the proposed protocol with its counterpart. The average packet loss can be calculated by subtracting number of data packet actually received by the base station from the total number of sent packet. The following formula used to calculate the number of lost packet:

Number of lost = (Number of sent packet) – (Number of received packet at base station)

7.2.2 Average End-to-End Delay

The average end-to-end delay is the time taken for transmitting a data packet (application packets) from source to the base station. This term has much significance in this kind of network because the multi-hop routing protocol is used.

The average end-to-end delay is calculated by taking the average delay for every application packet transmitted and received correctly by the base station.

*Packet delay (*ms*) = (Receive Time at sink node) – (Transmit Time from Source node)*
*Average packet delay (*ms*) = (Sum of all Packet Delays) / (Total Number of Received Packets)*

7.2.3 Energy Saving of Network Overall

This metric measures the energy saved by network overall at the end of the simulation time after summing the remaining energy of each node in the network. The Energy calculation is done by loading an initial energy into all nodes in the network (all nodes have the same initial energy), and setting other energy parameters such as receiving power, transmitting power, and idle power. The following formulas are used to calculate the saved energy:
*Total energy consumption from the network (*J*) = summation of remaining energy for all nodes*

7.2.4 Average Number of Hops

This metric measures the average number of hops needed to transmit data packets from source to sink node. The calculation is done by calculating the number of MAC layer packets sent divided by the number of application packets sent from sources nodes.

Average Number of hops = (MAC layer packets) / (Number of transmitted packets)

7.3 Simulation Results

It can be showed that the proposed TWELGossiping routing protocol perform better when compared to the other routing protocols. We investigated performance of the proposed protocol by comparing total number of Lost Packet, average end-end delay, total saved energy overall network and finally average number of hop count. The simulation flows as described in the following diagram (figure 7.3), For TCL and perl programing codes see appendix A3:

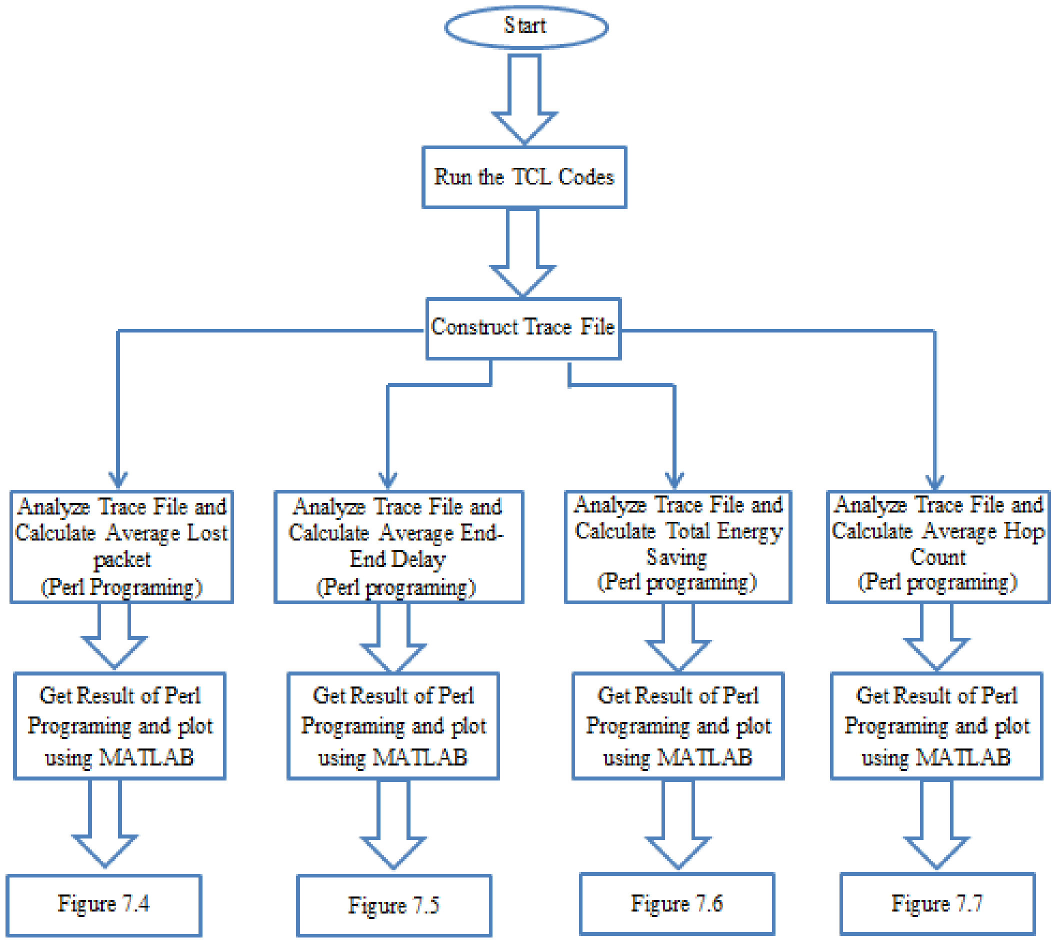

Figure 7.3 Block diagram describes Simulation flow

7.3.1 Total Number of Lost Packet

This metric shows the ability of a routing protocol to handle an increase in the network load. Figure 7.4 shows the results of the total number of lost Packets with increasing the number of sent packets. It can be seen that by increasing number of sent packet, the network begun to loss packet. This is natural because the network load becomes larger and there will high traffic especially in the nodes near to the base station. We can calculate total number of lost packet by using the equation given in section 7.2.1.

It is seen from figure 7.4 that when comparing to other protocols, TWELGossiping has minimum lost packet. This is because TWELGossiping can handle the high load in the network and also, this happens because the TWELGossiping routing protocol uses different paths to deliver data packets to the base station so there is no high load on any specific path.

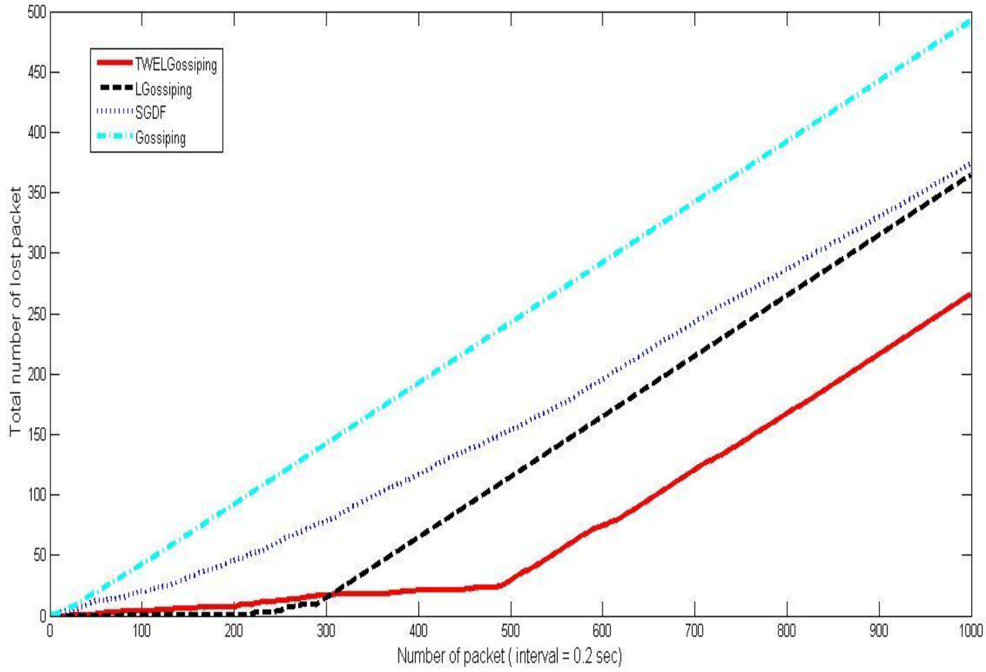

Figure 7.4 Simulation result of packet lost parameter

7.3.2 Average End-to-End Delay

This metric measures the time between a packet originating from the source node until the time that packet is received correctly at the base station. This metric also shows the ability of the routing protocol to deliver data packets in a shorter period of time compared with the counterpart protocols. In this simulation work, the average end-to-end delay was calculated in milliseconds (msec) by using the equation given in Secttion 7.2.2. Figure 7.5 illustrates the results of the four different routing protocols. The trend of all graphs is increase with increasing the number of packets in the network. This happens because the network load increases. Thus, the source node needs more time to deliver data packets to the sink node because of the high traffic especially in the nodes near to the base station.

Figure 7.5 shows that the TWELGossiping protocol achieves a higher performance than the others protocols it has the ability to find the shortest path to the base station by finding the nearest node to the base station from its neighbor nodes to choose it as the next hop node.

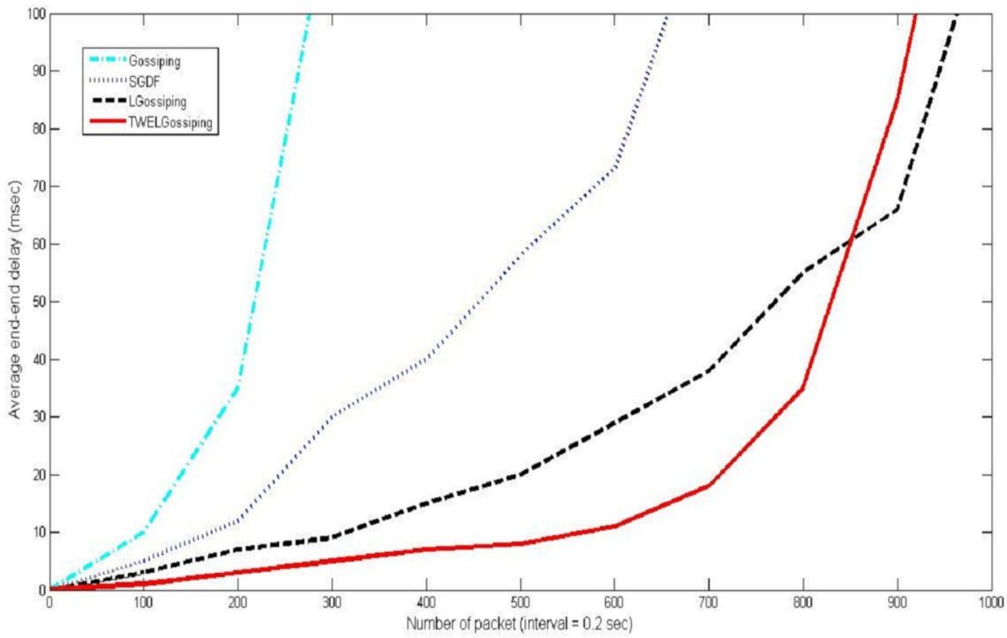

Figure 7.5 Simulation result of Average end-end delay

7.3.3 Total Energy Saving Overall Network

This metric is evaluated the amount of energy saved by the nodes during simulation time. At the beginning all nodes have the same initial energy (supposed) and to calculate the saved energy, we used the equation that given in section 7.3.3 which is:

Total saved energy overall network (J) = *summation of remaining energy of all nodes*

In the proposed scenario, nodes are deployed randomly to make the simulation more realistic. It can be seen from Figure 7.6 that our proposed protocol has saved more energy than the other protocols. This happened due to fairly using the energy of nodes (balancing the energy consumption). Furthermore, the relaying nodes are not selected randomly blindly. However, TWELGossiping chooses the nearest node to the base station to be the next hop for the packet. As a result, it produces a shorter path to the base station than the compared protocols.

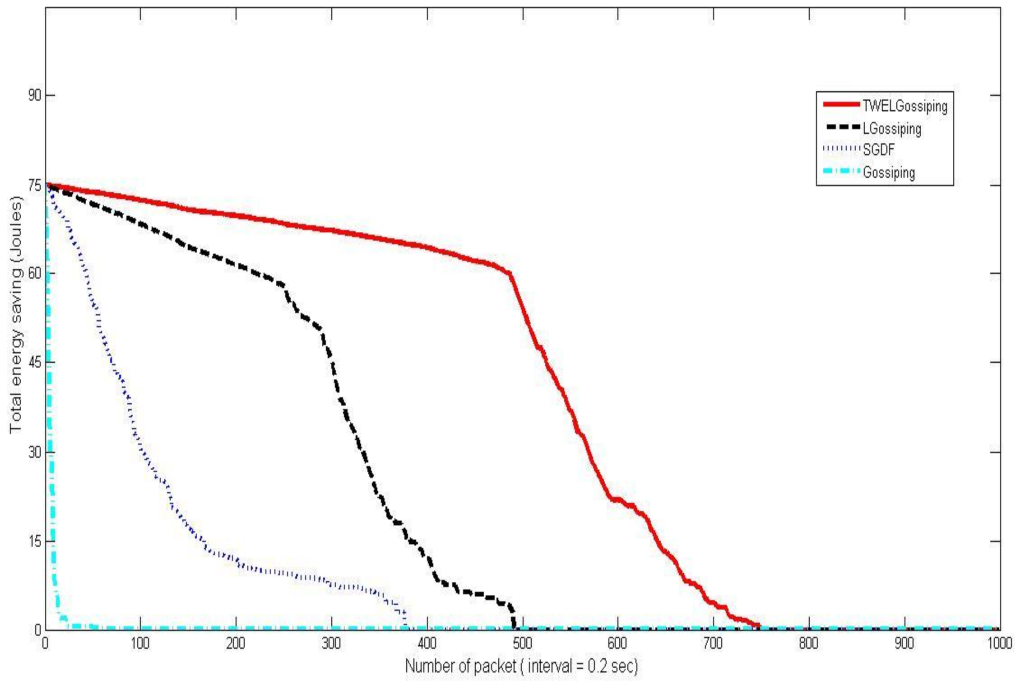

Figure 7.6 simulation result of total energy saving

7.3.4 Average Number of Hops

This metric shows the performance of routing protocol to deliver data using the minimum number of hops in a multi-hop system. In our test, the base station has been placed in the (0, 0) of *X* and *Y* axes and the sources is chosen randomly by the program so that we can measure the average hops from the source to the base station. Average number of hops is calculated by using the formula given in the section 7.2.4.

Figure 7.7 shows the results of testing this metric. The TWELGossiping reaches the base station in a less number of hops. The reason for this few hops is the TWELGossiping is the more effective protocol in choosing a route which has the minimum number of hops to the base station.

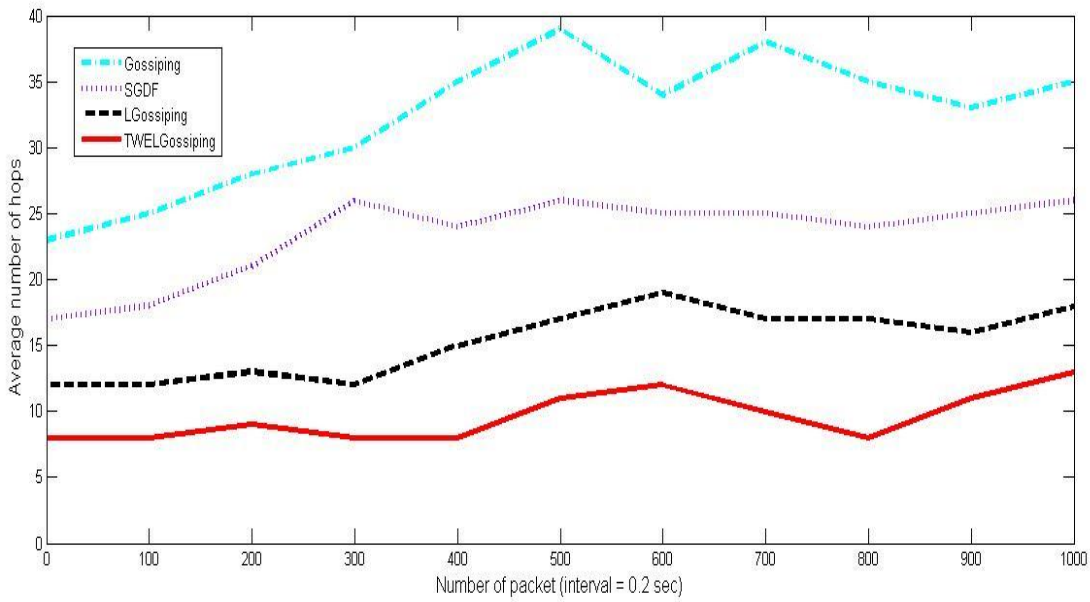

Figure 7.7 simulation result of Average number of hops

Finally, from the results of the simulation we can reach this conclusion: Since our protocol (TWELGossiping) maximizes network lifetime by saving energy and deliver the data in an efficient way with least delay, it is well suited for applications in which the energy, network lifetime and time delay are the main considerations.

CHAPTER EIGHT
CONCLUSION

This thesis is a result of a deep research in Wireless Sensor Networks. General information about WSNs, hardware and network architecture of sensor nodes, applications of WSNs and clustering and routing protocols has been investigated in this work. Since wireless sensor nodes are limited in energy, which is normally battery operated, efficient energy routing protocols must be used. The main objective of routing protocols is that must or try to minimize energy consumption and maximize the network life time. So, a wide comparison of some energy efficient routing protocols has been made.

After this detailed research about WSNs, we have developed one of these energy efficient routing protocols (LGossiping) and proposed the **T**wo **W**ay **E**fficient **L**ocation-based **G**ossiping protocol (TWELGossiping). Our proposed protocol has treated the main disadvantages of gossiping like delay, packet lost and energy consumption. It is consists of two phases; network setup phase and routing phase. In the setup phase, each node knows its distance or its depth from the base station by creating a gradient between the nodes and the base station. While In the second phase, routing phase, when a node detects an event, it sends this event to the base station through two ways. The nodes that contribute in the packet delivering process are chosen according to their distances and their residual energy, where the node that has more remaining energy and minimum distance to the base station will be selected as a next hop for the packet.

The performance of TWELGossiping routing protocol is evaluated by comparing it with the three routing protocols; Gossiping, LGossipig and SGDF on a simulation platform, where the delay, lost packet, total energy and live node per round are used as the performance metric to compare the proposed protocol with these three routing protocols. The simulation results of this work show that the TWELGossiping protocol achieves better performance than its counterparts routing protocols in terms of delay, packet delivery ratio and energy consumption (prolong the network lifetime). Our protocol reduces the delay by selecting the nearest node to the base station to be next hop for the packet. Moreover there is a possibility of being one way of these two ways faster than the other. It increases packet delivery ration by sending the packet through two ways, in case the packet fails to reach the base station from a way, it could be able to reach the base station through the second way. Finally, it increases the network lifetime by balancing and fairly using the energy of nodes, where the node that has more energy is chosen when sending the packet.

As a result, since our protocol (TWELGossiping) maximizes network lifetime by saving energy and deliver the data in an efficient way with least delay, it is well suited for applications in which the energy, network lifetime and time delay are the main considerations.

REFERENCES

Abbasi, AA., & Younis, M. (2007). A survey on clustering algorithms for wireless sensor networks. *Computer Communications Journal*, *30* (14), 2826–2841.

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless sensor networks: A survey. *Computer Networks (Elsevier) Journal*, 393–422.

Akyildiz, I. F., & Canvuran, M. (2010). *Wireless sensor networks.* Singapore by Markono Print Media Pte: WILEY.

Al_karaki J.N., & Kamal, A.E. (2004). Routing techniques in wireless sensor networks: A survey. *IEEE Wireless Communications*, *11* (6), 6 – 28.

Ammer, J., Rabaey, J., da Silva, J. L., & Patel, D. (2000). Picoradio: Ad-hoc wireless networking of ubiquitous low-energy sensor/monitor nodes. In *Proceedings of the IEEE Computer Society Annual Workshop on VLSI (WVLSI'00)*, Orlando, FL, USA, 9–12.

Banerjee, S., & Khuller, S. (2001). A clustering scheme for hierarchical control in multi-hop wireless networks. *In Proceedings of 20th Joint Conference of the IEEE Computer and Communications Societies (INFOCOM' 01), Anchorage, AK*, 2, 1028 – 1037.

Bandyopadhyay, S., & Coyle, E. (2003). An energy efficient hierarchical clustering algorithm for wireless sensor networks. *In Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003), San Francisco, California*, 3, 1713 – 1723.

Bani Yassein, M., Al-zou'bi, A., Khamayseh, Y., & Mardini, W. (2009). Improvement on LEACH protocol of wireless sensor network (VLEACH). *International Journal of Digital Content Technology and its Applications*. *3* (2), 132 – 136.

Bouabdallah, F., Bouabdallah, N., & Boutaba, R. (2009). On Balancing Energy Consumption in Wireless Sensor Networks. *IEEE Transactions on Vehicular Technology*, *58 (6)*, 2909 – 2924.

Boomerang shooter detection system (n.d). Last visit July 20, 2011, from http://bbn.com/boomerang.

Chakrabarti, D., & Seberry, J. (2006). Combinatorial structures for Design of Wireless Sensor Networks. *Applied Cryptology and Network Security Conference ACNS06*, Singapore, 365- 374.

Chintalapudi, K., Paek, J., Kothari, N., Rangwala, S., Caffrey, J., Govindan, R., et al. (2006). Monitoring civil structures with a wireless sensor network. *IEEE Internet Computing*, *10* (2), 26–34.

Chen, Y., & Zhao, Q. (2005). On the lifetime of wireless sensor networks. IEEE communications letters, 9 (11), 976-978.

CRUISE, WP112, D112.*1* (2006*). Report on WSN applications, their requirements, application-specific WSN issues and evaluation metrics*, December 2006, website: http://www.ist-cruise.eu.

Dechene, D. J., El Jardali, A., Luccini, M., & Sauer, A. (2006). A Survey of Clustering Algorithms for Wireless Sensor Networks. *Computer engineering*, 1 – 10.

Depienne, F. (2007). Wireless sensor networks application for agricultural environment sensing in developing countries. Semester Project Report of a Master Candidate in Communication Systems website: http://commonsense.epfl.ch/Admin/Documents/report_francois_depienne.pdf

Demirbas M., Arora, A., & Mittal, V. (2004). FLOC: a fast local clustering service for wireless sensor networks. *In Proceedings of Workshop on Dependability Issues in Wireless Ad Hoc Networks and Sensor Networks (DIWANS'04), Palazzo dei Congressi, Florence, Italy*, 1 – 6.

Ding, P., Holliday, J., & Celik, A. (2005). Distributed energy efficient hierarchical clustering for wireless sensor networks. *In Proceedings of the IEEE International Conference on Distributed Computing in Sensor Systems(DCOSS'05), Marina Del Rey, CA*, 322 – 339.

Eroglu, M. (2006). New stack architecture for sensor networks. *Dissertation*, Middle East Technical University.

e-SENSE, WP1, D1.2.1,(2006). *Scenarios and audio visual concepts*, September 2006, website: http://www.ist-esense.org

He, T., Krishnamurthy, S., Stankovic, J. A., Abdelzaher, T., Luo, L., Stoleru, R., et al. (2006). VigilNet: An integrated sensor network system for energy-efficient surveillance. *ACM Transactions on Sensor Networks*, *2* (1), 1–38.

Hedetniemi, S. M., Hedetniemi S. T., & Liestman, A. L. (1988). A survey of gossiping and broadcasting in communication networks. *Networks*, *18* (4), 319–349.

Heinzelman, W. B. (2000). Application-Specic Protocol Architectures for Wireless Networks. *Doctoral Dissertation, Massachusetts Institute of Technology*

Heinzelman, W. R., Chandrakasan, A., & Balakrishnan, H. (2000). Energy-efficient communication protocol for wireless microsensor networks. *In proceedings of the 33rd Hawaii International Conference on System Science,Maui, HI, USA*, *2* (1), 1–10.

Heinzelman, W.B., Chandrakasan, A.P., & Balakrishnan, H. (2002). Application specific protocol architecture for wireless microsensor networks. *IEEE Transactions on Wireless Networking, 1* (4), 660 – 670.

Hill, J., Szewcyk, R., Woo, A., Culler, D., Hollar, S., & Pister, K. (2000). System architecture directions for networked sensors. *In Proceeding ASPLOS-IX Proceedings of the ninth international conference on Architectural support for programming languages and operating systems, 28* (5), *1-18.*

Hoblos, G., Staroswiecki, M., & Aitouche, A. (2000). Optimal design of fault tolerant sensor networks. *In Proceedings of IEEE International Conference on Control Applications Anchorage, Alaska, US,* 467–472.

Huang, H., & Wu, J. (2005). A probabilistic clustering algorithm in wireless sensor networks. In *Proceeding of IEEE 62nd Semiannual Vehicular Technology Conference (VTC), Dallas,* 1796 – 1798.

Intanagonwiwat, C., Govindan, R., & Estrin, D. (2000). Directed diffusion: a scalable and robust communication paradigm for sensor networks. *MobiCom '00 Proceedings of the 6th annual international conference on Mobile computing and networking,* 56–67.

Ilyas, M., & Mahgoub, I. (Eds.). (2005). *Handbook of sensor networks: Compact wireless and wired sensing systems*. United States of America: CRC Press LLC.

Intanagonwiwat, C., Govindan, R., Estrin, D., Heidemann, J.,& Silva, F.(2003). Directed Diffusion for Wireless Sensor Networks. *IEEE/ACM Transactions on Networking*, *11* (1), 2–16.

Intel StrongARM SA-1100 (2000). Microprocessor Brief Data Sheet. *Intel product documentation*.

Juang, P., Oki, H., Wang, Y., Martonosi, M., Peh, L. S., & Rubenstein, D. (2002). Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with ZebraNet. *ACM SIGOPS Operating Systems Review*, *36* (5), 96–107.

Karl, H., & Willig, A. (2005). *Protocols and architectures for wireless sensor networks*. Great Britain by Antony Rowe: John Wiley & Sons.

Karlof, C., & Wanger, D. (2003). Secure routing in wireless sensor networks: Attacks and countermeasures. *Proceedings of the First IEEE. 2003 IEEE International Workshop on Sensor network protocols and applications*, 113 - 127.

Krishnamachari, B., Estrin, D., & Wicker, S. (2002). Modeling data centric routing in wireless sensor networks. In *Proceedings of IEEE INFOCOM, New York, NY*, 1 – 11.

Kheiri, S., Ghaznavi Goushchi, M.B., Rafiee, M., & Seyfe, B. (2009). An improved gossiping data distribution technique with emphasis on reliability and resource constraints. *IEEE 2009 International Conference on Communications and Mobile Computing*, 247 – 252.

Kim, Y., Schmid, T., Charbiwala, Z. M., Friedman, J., & Srivastava, M. B. (2008). NAWMS: Nonintrusive Autonomous Water Monitoring System. In *Proceedings of ACM SenSys'08 Raleigh, NC, USA*, 309–322.

Kredo II, K., & Mohapatra, P. (2007). Medium access control in wireless sensor networks. *The International Journal of Computer and Telecommunications Networking*, *51* (4), 961–994.

Lédeczi, Á., Nádas, A., Völgyesi, P., Balogh, G., Kusy, B., Sallai, J., et al. (2005). Countersniper system for urban warfare. *ACM Transactions on Sensor Networks*, *1* (2), 153–177.

Li, L. & Halpern, J. Y. (2001). Minimum energy mobile wireless networks revisited. In *Proceedings of the IEEE International Conference on Communications (ICC'01), Helsinki, Finland*, 1, 278 – 283.

Li, Y., Thai, M. T., & Wu, W. (Eds.). (2008). *Wireless Sensor Networks and Applications*. NewYork: Springer.

Li, S., Ma, X., Wang, X., & Tan, M. (2011). Energy-efficient multipath routing in wireless sensor network considering wireless interference. *J Control Theory Application*, *9* (1), 127–132.

Lindsey, S., Raghavendra, C. S., & Sivalingam, K. (2001). Data gathering in sensor networks using the energy*delay metric. In proceedings 15th International Parallel and Distributed Processing Symposium, 2001 – 2008.

Lindsey, S. & Raghavendra, C. S. (2002). PEGASIS: Power-Efficient Gathering in Sensor Information Networks. *Proceedings of the IEEE Aerospace Conference, Big Sky, MT, USA*, 3, 3-1125 - 3-1130.

Liu, W., & Yu, J. (2009). Energy efficient clustering and routing scheme for Wireless Sensor Networks. *IEEE International Conference on Intelligent Computing and Intelligent Systems, ICIS,* 612 – 616.

Loscri, V., Morabito, G., & Marano, S. (2005). A Two-Level Hierarchy for Low-Energy Adaptive Clustering Hierarchy. *IEEE 62nd Vehicular Technology Conference, 2005. VTC-2005-Fall*, 62, 1809 – 1813.

Mainwaring, A., Polastre, J., Szewczyk, R., Culler, D., & Anderson, J. (2002). Wireless Sensor Networks for Habitat Monitoring. *In Proceedings of the 1st ACM Workshop on Wireless Sensor Networks and Applications, Atlanta, GA*, September 2002.

Malan, D., Fulford-Jones, T., Welsh, M., & Moulton, S. (2004). CodeBlue: an Ad Hoc sensor network infrastructure for emergency medical care. In *Proceedings of Workshop on Applications of Mobile Embedded Systems (WAMES 2004),* 1 – 4.

Manjeshwar, A. & Agrawal, D. P. (2001). TEEN: a protocol for enhanced efficiency in wireless sensor networks. In *Proceedings of the 1st International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing, San Francisco, USA*, 2009 – 2015.

Manjeshwar, A. & Agrawal, D. P. (2002). APTEEN: a hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks. In *Proceedings of the 2nd International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing, Ft. Lauderdale, FL, USA*, 195 – 202.

Mhatre, V., Rosenberg, C., Kofman, D., Mazumdar, R., & Shroff , N. (2005). A minimum cost surveillance sensor network with a lifetime constraint. *IEEE Transactions on Mobile Computing, 4* (1), 4–15.

Nayak, A., & Stojmenovic, I. (Eds.). (2010). *Wireless Sensor and Actuator Networks Algorithms and Protocols for Scalable Coordination and Data Communication.* United State Of America: John WILEY & Sons.

Oh, H., & Chae, K. (2007). An energy-efficient sensor routing with low latency, scalabilit in wireless sensor networks. *International Conference on Multimedia and Ubiquitous Engineering (MUE'07),* 147 – 152.

Oh, H., Bahn, H., & Chae, K. (2005). An energy- efficient sensor routing scheme for home automation networks. *IEEE Transactions on Consumer Electronics, 51* (3), 836-839.

Paek, J., Chintalapudi, K., Cafferey, J., Govindan, R., & Masri, S. (2005). . In *Proceedings of the 2nd IEEE Workshop on Embedded Networked Sensors (EmNetS-II), Sydney, Australia*, 1 - 10.

Petriu, E. M., Georganas, N. D., Petriu, D. C., Makrakis, D., & Groza V. Z. (2000). Sensor-based information appliances. *Instrumentation & Measurement Magazine, 3* (4), 31–35.

Pathan, A.K., Hong, S. C., & Hyung, W. L. (2006). Smartening the environment using wireless sensor networks in a developing country. *Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference*, 705-709.

Rabaey, J. M., Ammer, M. J., da Silva J. L., Patel, D., & Roundy, S. (2000). Picoradio supports ad hoc ultra-low power wireless networking. *IEEE Computer, 33* (7): 42–48.

Raghunathan, V., Schurgers, C., Park, S., & Srivastava, M. B. (2002). Energy-Aware Wireless Microsensor Networks. *IEEE Signal Processing Magazine,* (19) 40–50.

Rudas, I. J., Fodor, J., & Kacprzyk, J. (Eds.). (2009). Towards Intelligent Engineering and Information Technology. Berlin: Springer

Sankarasubramaniam, Y., Akyildiz, I. F., & Mchughlin S . W. (2003). Energy Efficiency based Packet Size Optimization in Wireless Sensor Networks. *Proceedings of the First IEEE. 2003 IEEE International Workshop on Sensor Network Protocols and Applications,* 1-8.

Shen, C., Srisathapornphat, C., & Jaikaeo, C. (2001). Sensor information networking architecture and applications. *IEEE Personal Communications, 8* (4):52–59

Shih, E., Cho, S., Ickes, N., Min, R., Sinha, A., Wang, A., et al. (2001). Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks. *Proc* . ACM *MobiCom 01.* Rome. Italy, 272-286.

Sihavaran, T., Blair, G., Friday, A., Wu, M., Limon, H. D., Okanda, P., et al. (2004). Cooperating sentient vehicles for next generation automobiles. In *Proceeding of MobiSys 2004, 1st ACM Workshop on Applications of Mobile Embedded Systems*.

Slijepcevic, S., & Potkonjak , M. (2001). Power efficient organization of wireless sensor networks. *ICC 2001. IEEE International Conference on Communications,* (2) 472 – 476.

Sohrabi, K., Gao, J., Ailawadhi, V., & Pottie, G.J. (2000). Protocols for self-organization of a wireless sensor network. *IEEE Personal Communications*, *7* (5), 16–27.

Sohraby, K., Minoli, D., & Znati, T. (2007). Wireless sensor networks, technology, protocols, and applications. United States of America: John Wiley & Sons.

Swami, A., Zhao, Q., Hong, Y. W., & Tong, L. (Eds.). (2007). *Wireless sensor networks signal processing and communications perspectives*. Great Britain by Antony Rowe: John Wiley & Sons.

Verdone, R. Dardari, D. Mazzini, G, & Conti, A. (2007). *Wireless sensor and actuator networks technology, analysis and design.* Great Britain: Elsevier.

Wang, B., (2010). *Coverage Control in Sensor Networks*. United Kingdom: Springer-Verlag London.

Werner-Allen, G., Lorincz, K., Ruiz, M., Marcillo, O., Johnson, J., Lees, J., et al. (2006). Deploying a wireless sensor network on an active volcano. *IEEE Internet Computing*, *10* (2), 18–25.

Wilson, J. S. (Ed.). (2005). *Sensor technology handbook*. USA: Elsevier

Xiangning, F., & Yulin, S. (2007). Improvement on LEACH protocol of wireless sensor network. *International Conference on Sensor Technologies and Applications, 2007. SensorComm,* 260 – 264.

Ye, M., Li, C., Chen, G., & Wu, J. (2007). EECS: An Energy Efficient Clustering Scheme in Wireless Sensor Networks. IPCCC 2005. 24th IEEE International Performance, Computing, and Communications Conference, 535 – 540.

Yen, W., Chen, C.-W., & Yang, C.-H. (2008). Single gossiping with directional flooding routing protocol in wireless sensor networks. *3rd IEEE Conference on Industrial Electronics and Applications, 2008. ICIEA 2008, 40* (3), 1604 - 1609.

Younis, O., Krunz, M., & Ramasubramanian, S. (2006). Node Clustering in Wireless Sensor Networks: Recent Developments and Deployment Challenges. *IEEE Network*, *20* (3), 20 – 25.

Younis, O. & Fahmy, S. (2004). Distributed clustering in ad-hoc sensor networks: a hybrid, energy-efficient approach. *IEEE Transactions on Mobile Computing*, *3* (4):366–379.

Zhang, P., Sadler, C., Lyon, S., & Martonosi, M. (2004). Hardware design experiences in ZebraNet. *Proceedings of (ACM) the 2nd international conference on Embedded networked sensor systems SenSys 04*, 227-238.

Zheng, J., & Jamalipour, A. (Eds.). (2009). *Wireless sensor networks a networking perspective*. United State of America: John Wiley & Sons.

Zhou, H., Jiang, Z., & Xiaoyan, M, (2006). Study and design on cluster routing protocols of wireless sensor networks. *Dissertation.*

Zhang, Y., & Cheng, L. (2004). Flossiping: A new routing protocol for wireless sensor networks. *Proceedings of the 2004 IEEE International Conference on Networking, Sensing & Control Taipei, Taiwan*, 1218 – 1223.

APPENDICES

LIST OF SYMBOLS/ABBREVIATIONS

WSNs	Wireless Sensor Networks
TWELGossiping	Two Way Efficient Location-based Gossiping
MEMS	Micro Electro-Mechanical Systems
GPS	Global Position System
RF	Radio Frequency
HWSN	Hierarchical Wireless Sensor Networks
DWSN	Distributed Wireless Sensor Networks
ISM	Industrial, Scientific and Medical
IEEE	The Institute of Electrical and Electronics Engineers
LAN	Local Area Network
PDA	Personal Digital Assistant
RAM	Random Access Memory
MAC	Medium Access Control
QoS	Quality of Service
CPU	Central Processor Unit
ROM	Read Only Memory
EEPROM	Electrically Erasable Programmable ROM
CSI	Channel State Information
REI	Residual Energy Information

UMTS	Universal Mobile Telecommunications System
CSN	COMMON-Sense Net (a project)
C4ISRT	Command, Control, Communications, Computing, Intelligence, Surveillance, Reconnaissance, and Targeting
NBC	Nuclear, Biological, and Chemical
DSP	Digital Signal Processing
EKG	ElectroCardioGraph
EMG	ElectroMyoGraph
GUI	Graphical User Interface
DVD	Digital Versatile Disc
NAWMS	Nonintrusive Autonomous Water Monitoring System
SHM	Structural Health Monitoring
RFID	Radio Frequency Identification
AMR	Automatic Meter Reading
HVAC	Heating, Ventilating, and Air-Conditioning
CH	Cluster Head
CM	Cluster Member
BS	Base Station
LCA	Linked Cluster Algorithm
RCC	Random Competition based Clustering
BFS	Breadth-First-Search
EEHC	Energy Efficient Hierarchical Clustering
LEACH	Low Energy Adaptive Clustering Hierarchy
TLLEACH	Two-Level Hierarchy LEACH
E-LEACH	Energy-LEACH
M-LEACH	Multi-hop-LEACH

LEACH-C	Centralized LEACH
V-LEACH	new Version of LEACH
EECS	Energy Efficient Clustering Scheme
PEGASIS	Power-Efficient GAthering in Sensor Information Systems
FLOC	Fast Local Clustering service
HEED	Hybrid Energy-Efficient Distributed Clustering
SPIN	Sensor Protocols for Information via Negotiation
TEEN	Threshold sensitive Energy Efficient sensor Network protocol
APTEEN	Adaptive Threshold-sensitive Energy-Efficient sensor Network
HT	Hard Threshold
ST	Soft threshold
EECR	Energy-Efficient Clustering and Routing Protocol
MECN	Minimum Energy Communication Network
SMECN	Small MECN
PRADA	Probe-Based Distributed Protocol for Knowledge Range Adjustment
PTKF	Partial Topology Knowledge Forwarding
SAR	Sequential Assignment Routing
TDMA	Time Division Multiple Access
CDMA	Code Division Multiple Access
EESR	Energy-Efficient Sensor Routing
FLOSSIPING	Flooding Gossiping
SGDF	Single Gossiping with Directional Flooding
LGossiping	Location-based Gossiping

LIST OF FIGURES AND TABLES

TCL CODES

```
#####################################################
#     Routing TWELGossiping over 802.15.4                  #
#             (beacon enabled)                             #
#       Copyright (c) 2003 Samsung/CUNY                    #
# - - - - - - - - - - - - - - - - - - - - -------------------------------- - - - #
#          Prepared by Jianliang Zheng                     #
#           (zheng@ee.ccny.cuny.edu)                       #
#####################################################

# ===========================================================
# Define options
# ===========================================================
set val(chan)          Channel/WirelessChannel    ;# Channel Type
set val(prop)          Propagation/TwoRayGround   ;# radio-propagation model
set val(netif)         Phy/WirelessPhy/802_15_4
set val(mac)           Mac/802_15_4
set val(ifq)           Queue/DropTail/PriQueue    ;# interface queue type
set val(ll)            LL                         ;# link layer type
set val(ant)           Antenna/OmniAntenna        ;# antenna model
set val(ifqlen)        100                        ;# max packet in ifq
set val(nn)            100                        ;# number of nodes
set val(rp)            twel                       ;# TWELGossiping
set val(x)                 100
set val(y)                 100

set val(tr)                wpan_demo3.tr
##set val(nam)             wpan_demo3.nam
set val(nodeDown)          no
#set val(errRate)          0                      ;# %
set val(traffic) cbr                              ;# mix/cbr/poisson/ftp
set val(trInterval)        0.2                    ;# in seconds

#read command line arguments
proc getCmdArgu {argc argv} {
	global val
	for {set i 0} {$i < $argc} {incr i} {
		set arg [lindex $argv $i]
		if {[string range $arg 0 0] != "-"} continue
		set name [string range $arg 1 end]
		set val($name) [lindex $argv [expr $i+1]]
	}
}
getCmdArgu $argc $argv

set appTime1               50.5     ;# in seconds
```

```
set appTime2            53.8    ;# in seconds
set appTime3            55.8    ;# in seconds
set appTime4            58.8    ;# in seconds
set appTime5            59.8    ;# in seconds
set stopTime          500       ;# in seconds


# Initialize Global Variables
set ns_          [new Simulator]
set tracefd     [open ./$val(tr) w]
$ns_ trace-all $tracefd
if { "$val(nam)" == "wpan_demo3.nam" } {
        set namtrace     [open ./$val(nam) w]
        $ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
    }

$ns_ puts-nam-traceall {# nam4wpan #}                  ;# inform nam that this is a trace
file for wpan (special handling needed)

Mac/802_15_4 wpanCmd verbose on
Mac/802_15_4 wpanNam namStatus on                 ;# default = off (should be
turned on before other 'wpanNam' commands can work)
#Mac/802_15_4 wpanNam ColFlashClr gold                    ;# default = gold
#Mac/802_15_4 wpanNam NodeFailClr grey                    ;# default = grey

# For model 'TwoRayGround'
set dist(5m)  7.69113e-06
set dist(9m)  2.37381e-06
set dist(10m) 1.92278e-06
set dist(11m) 1.58908e-06
set dist(12m) 1.33527e-06
set dist(13m) 1.13774e-06
set dist(14m) 9.81011e-07
set dist(15m) 8.54570e-07
set dist(16m) 7.51087e-07
set dist(20m) 4.80696e-07
set dist(25m) 3.07645e-07
set dist(30m) 2.13643e-07
set dist(35m) 1.56962e-07
set dist(40m) 1.20174e-07
Phy/WirelessPhy set CSThresh_ $dist(20m)
Phy/WirelessPhy set RXThresh_ $dist(20m)

# set up topography object
set topo       [new Topography]
$topo load_flatgrid $val(x) $val(y)

# Create God
```

```
set god_ [create-god $val(nn)]

set chan_1_ [new $val(chan)]

# configure node

$ns_ node-config -adhocRouting $val(rp) \
			-llType $val(ll) \
			-macType $val(mac) \
			-ifqType $val(ifq) \
			-ifqLen $val(ifqlen) \
			-antType $val(ant) \
			-propType $val(prop) \
			-phyType $val(netif) \
			-topoInstance $topo \
			-agentTrace ON \
			-routerTrace ON \
			-macTrace ON \
			-movementTrace OFF \
		-channel $chan_1_ \
			-energyModel "EnergyModel" \
		-initialEnergy 100 \
		-rxPower 0.075 \
		-txPower 0.075 \
			-idlePower 0.0003 \
		-sleepPower 0.0 \




for {set i 0} {$i < $val(nn) } {incr i} {
	set node_($i) [$ns_ node]
	$node_($i) random-motion 0		;# disable random motion
}


 BlockSize = $BlockSize, CSkip0 = $cskip0, CSkip1 = $cskip1, CSkip2 = $cskip2, CSkip3 = $cskip3, CSkip4 = $cskip4, CSkip5 = $cskip5, CSkip6 = $cskip6\""

source ./wpan_100.scn

$ns_ at 0.0	"$node_(0) NodeLabel \"base station\""

$ns_ at 0.0	"$node_(0) sscs startCTPANCoord 1 3 3 UseClusterTree 7 4 4"	;# startCTPANCoord <txBeacon=1> <BO=3> <SO=3>
```

```
$ns_ at 0.5	"$node_(1) sscs startCTDevice"	;# startCTDevice <isFFD=1>
<assoPermit=1> <txBeacon=0> <BO=3> <SO=3>
$ns_ at 1.8	"$node_(2) sscs startCTDevice"
$ns_ at 2.3	"$node_(3) sscs startCTDevice"
$ns_ at 2.8	"$node_(4) sscs startCTDevice"

$ns_ at 3.5	"$node_(5) sscs startCTDevice 1 1 1"
$ns_ at 5.4	"$node_(6) sscs startCTDevice 1 1 1"
$ns_ at 5.5	"$node_(7) sscs startCTDevice 1 1 1"
$ns_ at 6.6	"$node_(8) sscs startCTDevice 1 1 1"

$ns_ at 7.3	"$node_(9) sscs startCTDevice"
$ns_ at 8.8	"$node_(10) sscs startCTDevice"
$ns_ at 9.3	"$node_(11) sscs startCTDevice"
$ns_ at 10.8	"$node_(12) sscs startCTDevice"

# ---

for {set i 13} {$i < 17} {incr i} {
	$ns_ at 11.3	"$node_($i) sscs startCTDevice"
}
for {set i 17} {$i < 21} {incr i} {
	$ns_ at 12.3	"$node_($i) sscs startCTDevice"
}
for {set i 21} {$i < 25} {incr i} {
	$ns_ at 13.3	"$node_($i) sscs startCTDevice"
}

# ---

for {set i 25} {$i < 29} {incr i} {
	$ns_ at 14.3	"$node_($i) sscs startCTDevice"
}
for {set i 29} {$i < 33} {incr i} {
	$ns_ at 15.3	"$node_($i) sscs startCTDevice"
}
for {set i 33} {$i < 41} {incr i} {
	$ns_ at 16.3	"$node_($i) sscs startCTDevice"
}

# ---

for {set i 41} {$i < 49} {incr i} {
	$ns_ at 17.3	"$node_($i) sscs startCTDevice"
}
for {set i 49} {$i < 57} {incr i} {
	$ns_ at 18.3	"$node_($i) sscs startCTDevice"
}
```

```
for {set i 57} {$i < 61} {incr i} {
	$ns_ at 19.3	"$node_($i) sscs startCTDevice"
}

# ---

for {set i 61} {$i < 65} {incr i} {
	$ns_ at 20.3	"$node_($i) sscs startCTDevice "
}
for {set i 65} {$i < 73} {incr i} {
	$ns_ at 21.3	"$node_($i) sscs startCTDevice"
}
for {set i 73} {$i < 77} {incr i} {
	$ns_ at 22.3	"$node_($i) sscs startCTDevice "
}
for {set i 77} {$i < 80} {incr i} {
	$ns_ at 23.3	"$node_($i) sscs startCTDevice"
}
for {set i 80} {$i < 85} {incr i} {
	$ns_ at 24.3	"$node_($i) sscs startCTDevice "
}

# ---

for {set i 85} {$i < 93} {incr i} {
	$ns_ at 26.3	"$node_($i) sscs startCTDevice"
}
for {set i 93} {$i < 101} {incr i} {
	$ns_ at 27.3	"$node_($i) sscs startCTDevice"
}


Mac/802_15_4 wpanNam PlaybackRate 12ms
$ns_ at $appTime1 "Mac/802_15_4 wpanNam PlaybackRate 1.0ms"
$ns_ at [expr $appTime1 + 0.5] "Mac/802_15_4 wpanNam PlaybackRate 2.0ms"

$ns_ at $appTime1 "puts \"\nTransmitting data ...\n\""

# Setup traffic flow between nodes

proc cbrtraffic { src dst interval starttime } {
   global ns_ node_
   set udp_($src) [new Agent/UDP]
   eval $ns_ attach-agent \$node_($src) \$udp_($src)
   set null_($dst) [new Agent/Null]
   eval $ns_ attach-agent \$node_($dst) \$null_($dst)
   set cbr_($src) [new Application/Traffic/CBR]
   eval \$cbr_($src) set packetSize_ 80
```

```
  eval \$cbr_($src) set interval_ $interval
  eval \$cbr_($src) set random_ 0
  #eval \$cbr_($src) set maxpkts_ 10000
  eval \$cbr_($src) attach-agent \$udp_($src)
  eval $ns_ connect \$udp_($src) \$null_($dst)
  $ns_ at $starttime "$cbr_($src) start"
}

proc poissontraffic { src dst interval starttime } {
  global ns_ node_
  set udp($src) [new Agent/UDP]
  eval $ns_ attach-agent \$node_($src) \$udp($src)
  set null($dst) [new Agent/Null]
  eval $ns_ attach-agent \$node_($dst) \$null($dst)
  set expl($src) [new Application/Traffic/Exponential]
  eval \$expl($src) set packetSize_ 70
  eval \$expl($src) set burst_time_ 0
  eval \$expl($src) set idle_time_ [expr $interval*1000.0-70.0*8/250]ms   ;#
idle_time + pkt_tx_time = interval
  eval \$expl($src) set rate_ 250k
  eval \$expl($src) attach-agent \$udp($src)
  eval $ns_ connect \$udp($src) \$null($dst)
  $ns_ at $starttime "$expl($src) start"
}

if {$val(rp) == "twel"} {
        Mac/802_15_4 wpanCmd callBack 2;# 0=none; 1=failure only (default);
2=both failure and success
}

if { ("$val(traffic)" == "mix") || ("$val(traffic)" == "cbr") || ("$val(traffic)" ==
"poisson") } {
  if { "$val(traffic)" == "mix" } {
        set trafficName "cbr + poisson"
        set traffic1 cbr
        set traffic2 poisson
  } else {
        set trafficName $val(traffic)
        set traffic1 $val(traffic)
        set traffic2 $val(traffic)
  }
  puts "\nTraffic: $trafficName"
  #Mac/802_15_4 wpanCmd ack4data on
  puts [format "Acknowledgement for data: %s" [Mac/802_15_4 wpanCmd
ack4data]]


   set bb [expr  {int (rand() * 100)}]            # choose source randomly
```

```
  ${traffic1}traffic $bb 1 $val(trInterval) $appTime1



  if { "$val(nodeDown)" == "yes" } {
          $ns_ at [expr $appTime2 + 3.0] "$node_(0) node-down"
          set tmpTime [format "%.1f" [expr $appTime2 + 3.0]]
          $ns_ at [expr $appTime2 + 3.0] "$ns_ trace-annotate \"(at $tmpTime) node
down: 0\""
          $ns_ at [expr $appTime2 + 10.0] "$node_(0) node-up"
          set tmpTime [format "%.1f" [expr $appTime2 + 10.0]]
          $ns_ at [expr $appTime2 + 10.0] "$ns_ trace-annotate \"(at $tmpTime) node
up: 0\""
  }
  Mac/802_15_4 wpanNam FlowClr -p AODV -c tomato
  Mac/802_15_4 wpanNam FlowClr -p ARP -c green
  Mac/802_15_4 wpanNam FlowClr -p MAC -c navy
  $ns_ at $appTime1 "$node_(67) add-mark m1 blue circle"
  $ns_ at $appTime1 "$node_(45) add-mark m2 blue circle"
  $ns_ at $appTime1 "$ns_ trace-annotate \"(at $appTime1) $traffic1 traffic from
node 67 to node 45\""
  $ns_ at $appTime2 "$node_(71) add-mark m3 green4 circle"
  $ns_ at $appTime2 "$node_(74) add-mark m4 green4 circle"
  $ns_ at $appTime2 "$ns_ trace-annotate \"(at $appTime2) $traffic2 traffic from
node 71 to node 74\""
  if { "$val(traffic)" == "cbr" } {
          set pktType cbr
          set pktType2 cbr
  } elseif { "$val(traffic)" == "poission" } {
          set pktType exp
          set pktType2 exp
  } else {
          set pktType cbr
          set pktType2 exp
  }
  Mac/802_15_4 wpanNam FlowClr -p $pktType -s 67 -d 45 -c blue
  Mac/802_15_4 wpanNam FlowClr -p $pktType2 -s 71 -d 74 -c green4
}

proc ftptraffic { src dst starttime } {
  global ns_ node_
  set tcp($src) [new Agent/TCP]
  eval \$tcp($src) set packetSize_ 60
  set sink($dst) [new Agent/TCPSink]
  eval $ns_ attach-agent \$node_($src) \$tcp($src)
  eval $ns_ attach-agent \$node_($dst) \$sink($dst)
```

```
   eval $ns_ connect \$tcp($src) \$sink($dst)
   set ftp($src) [new Application/FTP]
   eval \$ftp($src) attach-agent \$tcp($src)
   $ns_ at $starttime "$ftp($src) start"
}

if { "$val(traffic)" == "ftp" } {
   puts "\nTraffic: ftp"
   #Mac/802_15_4 wpanCmd ack4data off
   puts [format "Acknowledgement for data: %s" [Mac/802_15_4 wpanCmd
ack4data]]
   Mac/802_15_4 wpanNam FlowClr -p AODV -c tomato
   Mac/802_15_4 wpanNam FlowClr -p ARP -c green
   Mac/802_15_4 wpanNam FlowClr -p MAC -c navy
   $ns_ at $appTime1 "$node_(67) add-mark m1 blue circle"
   $ns_ at $appTime1 "$node_(45) add-mark m2 blue circle"
   $ns_ at $appTime1 "$ns_ trace-annotate \"(at $appTime1) ftp traffic from node 67
to node 45\""
   Mac/802_15_4 wpanNam FlowClr -p tcp -s 67 -d 45 -c blue
   Mac/802_15_4 wpanNam FlowClr -p ack -s 45 -d 67 -c blue
   $ns_ at $appTime2 "$node_(71) add-mark m3 green4 circle"
   $ns_ at $appTime2 "$node_(74) add-mark m4 green4 circle"
   $ns_ at $appTime2 "$ns_ trace-annotate \"(at $appTime2) ftp traffic from node 71
to node 74\""
   Mac/802_15_4 wpanNam FlowClr -p tcp -s 71 -d 74 -c green4
   Mac/802_15_4 wpanNam FlowClr -p ack -s 74 -d 71 -c green4
}

# defines the node size in nam
for {set i 0} {$i < $val(nn)} {incr i} {
        $ns_ initial_node_pos $node_($i) 2
}

# Tell nodes when the simulation ends
for {set i 0} {$i < $val(nn) } {incr i} {
    $ns_ at $stopTime "$node_($i) reset";
}

$ns_ at $stopTime "stop"
$ns_ at $stopTime "puts \"\nNS EXITING...\n\""
$ns_ at $stopTime "$ns_ halt"

proc stop {} {
    global ns_ tracefd appTime val env
    $ns_ flush-trace
    close $tracefd
    set hasDISPLAY 0
    foreach index [array names env] {
```

```
        #puts "$index: $env($index)"
        if { ("$index" == "DISPLAY") && ("$env($index)" != "") } {
                set hasDISPLAY 1
        }
    }
    if { ("$val(nam)" == "wpan_demo3.nam") && ("$hasDISPLAY" == "1") } {
            exec nam wpan_demo3.nam &
            }
}

puts "\nStarting Simulation..."
$ns_ run
```

PERL CODES

```
#!/usr/bin/perl

$infile='./test.tr';
print "$argc \n";
if ($#ARGV == 0)
{
 $infile=$ARGV[0];
 print "$infile \n"
}
$G=0;
$S=0;
$D=0;
$Gmac=0;
$forward=0;
$sendno=0;
$drop=0;
$Time_simulation=500;
my @PktNumber=();
my @PktTime=();

$Size=80;
$tmp=0;
open(DATA, "<$infile");
while(<DATA>)
{
  @x = split(' ');

  if ($x[1] >=40.0)
  {
```

```
if ($x[0] eq 'f')
  { $forward = $forward+1;
  }
if($x[0] eq 'D')
  { $drop=$drop+1;
   }
 if (($x[0] eq 's') && ($x[3] ne 'AGT'))
    { $sendno=$sendno+1;
    }
 if ($x[0] eq 's')
 {
    #print "send paquet \n";
        if ($x[3] eq 'AGT')
        {

          $G=$G+$Size;   # Calcul de G



          $exist=1;
          $nombre=$#PktNumber+1;
         # print "nombre :$nombre \n";
          for($i = 0; $i < $nombre ; $i++)
          {
             if ($x[5] == $PktNumber[$i]) {$exist=0;}
            # print "test for existence \n";
          }
          #print "exist : $exist \n";
          # vrai travail, pour le calcul de D
          if ($exist == 1)
          {
            push(@PktNumber, $x[5]);
            push(@PktTime, $x[1]);
            #print " Nombre de paquets après push :  $#PktNumber \n" ;

          }

        }
   else
        {
          ###print "supposed mac trame \n";
          if ($x[6] eq 'cbr' ) { $Gmac=$Gmac+$Size;}
    }
 }
 if ($x[0] eq 'r')
 {
    #print "receive packet \n ";
        if (($x[3] eq 'AGT') && ($x[2] eq '_20_'))
```

```
        {
        #if (($x[2] eq '_15_') || ($x[2] eq '_8_') ||($x[2] eq '_36_') ||($x[2] eq '_29_')
||($x[2] eq '_42_'))
# || ($x[2] eq '_42_') ||($x[2] eq '_32_') || ($x[2] eq '_35_') || ($x[2] eq '_68_') || ($x[2]
eq '_45_'))
# || ($x[2] eq '_25_') || ($x[2] eq '_77_') || ($x[2] eq '_66_') || ($x[2] eq '_27_') || ($x[2]
eq '_44_') || ($x[2] eq '_7_') || ($x[2] eq '_49_') || ($x[2] eq '_8_') || ($x[2] eq '_68_') ||
($x[2] eq '_24_') || ($x[2] eq '_57_') || ($x[2] eq '_42_') || ($x[2] eq '_12_') || ($x[2] eq
'_58_') || ($x[2] eq '_79_'))

                #print "AGT and 0 packet \n";
                $S=$S+$Size; #calcul de S
                #calcul de D
                $test=1;
                $i=0;
                #print " Nombre de paquets:  $#PktNumber \n" ;
                $nombre=$#PktNumber+1;
                while(($i<=$nombre) && ($test == 1))
                {
                  #print " $PktNumber[$i] - $x[5] \n";

  if ($PktNumber[$i] == $x[5])
                  {

                        $D=$D+$x[1]-$PktTime[$i];
                        splice(@PktNumber,$i,1);
                        splice(@PktTime,$i,1);
                 $nbreP++;
                        $test=0;
                        #print " un calcul doit se faire ici \n";
                  }
                  #print "boucle while \n";
                  $i++;
                }

      }
    }

  }


}
$out_file='./normalize_load.gp';
open(result,">>$out_file");
print result "\n";
print "tmp = $tmp \n";
$t=(($G/500)/250000);           # normalize load
print result "$t ";
```

```
$out_file='./normalize_throughput.gp';
open(result,">>$out_file");
print result "\n";
$t=(($S/500)/250000);           #normalize thruoghput
print result "$t ";
$out_file='./Average_end_to_end.gp';
open(result,">>$out_file");
print result "\n";
$t=($D*1000/$nbreP);    #average end-to-end-delay
print result "$t ";
$out_file='./mac_load.gp';
open(result,">>$out_file");
print result "\n";
$Ps=$S/$Gmac;
print result "$Ps ";
$out_file='./packet_lost.gp';
open(result,">>$out_file");
print result "\n";
$t=(($G/500)-($S/500))/80; #packet lost in packet/sec
print result "$t ";
$out_file='./delivary_raito.gp';
open(result,">>$out_file");
print result "\n";
$t=($S/$G)*100;               #delivary ratio throughput/load
print result "$t ";
$out_file='./routing_overhead.gp';
open(result,">>$out_file");
print result "\n";
$t=($forward/$sendno); #routing overhead no off packet /no of sent packet
print result "$t ";
$out_file='./Average_throughput.gp';
open(result,">>$out_file");
print result "\n";
$t=($S/500);               #Average throughput bit/sec
print result "$t ";
$out_file='./Average_load.gp';
open(result,">>$out_file");
print result "\n";
$t=($G/500);               #Average load bit/sec
print result "$t ";
$out_file='./number_Drop.gp';
open(result,">>$out_file");
print result "\n";
$t=$drop;               #number of drop packets
print result "$t ";
$out_file='./number_forward.gp';
open(result,">>$out_file");
print result "\n";
```

```
$t=$forward;                #number of forward packets
print result "$t ";
$out_file='./number_contol_packet.gp';
open(result,">>$out_file");
print result "\n";
$t=$sendno;                #number of contol packets
print result "$t ";


#!/usr/usc/bin/perl

#($#ARGV ne 0) || die "[USAGE] cbr-throughput.pl trace_file\n";

printf "%s\n", $ARGV[0];
open (INPUT, "<$ARGV[0]");
open (OUTPUT, ">sink_received");

@conn="";
$dest=-1;
$source=-1;
$sport = -1;
$dport = -1;
$seq=-1;

$num_received=0;
$num_dropped=0;
$num_send=0;

@srcid="";
@dstid="";
$num_conn_recv=0;
$num_conn_send=0;

@energy="";
@nodeid=(_0_,_1_,_2_,_3_,_4_,_5_,_6_,_7_,_8_,_9_,_10_,_11_,_12_,_13_,_14_,_
15_,_16_,_17_,_18_,_19_,_20_,_21_,_22_,_23_,_24_,_25_,_26_,_27_,_28_);
#,_29_,_30_,_31_,_32_,_33_,_34_,_35_,_36_,_37_,_38_,_39_,_40_,_41_,_42_,_43_
,_44_,_45_,_46_,_47_,_48_,_49_,_50_,_51_,_52_,_53_,_54_,_55_,_56_,_57_,_58_,
_59_,_60_,_61_,_62_,_63_,_64_,_65_,_66_,_67_,_68_,_69_,_70_,_71_,_72_,_73_,_
74_,_75_,_76_,_77_,_78_,_79_,_80_,_81_,_82_,_83_,_84_,_85_,_86_,_87_,_88_,_8
9_,_90_,_91_,_92_,_93_,_94_,_95_,_96_,_97_,_98_,_99_,_100_);

while (<INPUT>)
{
        next if !length($_);
        @things = split;

        #if( ($things[0] =~ /N/) && ($things[2]=~ /51.000/) )
```

```
#	{
		$source = $things[2] ;
		$energy[$source] = $things[13] ;
#	}
}


$total_send=0;
$total_recv=0;
$total_energy=0 * ($#nodeid+1);


foreach $nodeid (@nodeid)
{
	$total_energy += $energy[$nodeid];


}




close INPUT;
close OUTPUT;

$out_file='./energy_Saving.gp';
open(result,">>$out_file");
print result "\n";
print result "$total_energy";


#!/usr/bin/perl

$infile='./test.tr';
print "$argc \n";
if ($#ARGV == 0)
{
 $infile=$ARGV[0];
 print "$infile \n"
}
$G=0;
$S=0;
$D=0;
$Gmac=0;
$hop=0;
$gg=0;
$count=0;
$forward=0;
$sendno=0;
```

```
$drop=0;
$bb=0;
$cbrbyte=0;
$Time_simulation=500;
$counter=0;
my @PktNumber=();
#my @PktTime=();
# Mostafa
my @NodID=();
#my @two= [][];

$Size=80;
$tmp=0;
open(DATA, "<$infile");
while(<DATA>)
{
  @x = split(' ');

  if (($x[0] eq 's') && ($x[3] eq 'MAC') && ($x[6] eq 'cbr'))
     {
        $cbrbyte= $cbrbyte + $x[7];
     }
   if (($x[0] eq 's') && ($x[3] eq 'AGT'))
       {
        $counter=$counter + $Size;
       }
}
$out_file='./AveHop.gp';
open(result,">>$out_file");
print result "\n";
#print "tmp = $tmp \n";
$t=($cbrbyte/$counter);
print result "$t ";
```