CONSOLIDATE AND ADVANCE: AN EFFICIENT QOS MANAGEMENT TECHNIQUE IN HETEROGENEOUS WIRELESS SENSOR NETWORKS

By

D.PALANIVEL RAJAN

Reg No: 0720108012

of

KUMARAGURU COLLEGE OF TECHNOLOGY
COIMBATORE – 641 006

(An Autonomous Institution affiliated to Anna University, Coimbatore)

A PROJECT REPORT

Submitted to the

FACULTY OF INFORMATION AND COMMUNICATION ENGINEERING

In partial fulfillment of the requirements
for the award of the degree
Of

MASTER OF ENGINEERING
IN

COMPUTER SCIENCE AND ENGINEERING

MAY 2009

BONAFIDE CERTIFICATE

Certified that this project report titled "**CONSOLIDATE AND ADVANCE: AN EFFICIENT QOS MANAGEMENT TECHNIQUE IN HETEROGENEOUS WIRELESS SENSOR NETWORKS**" is the bonafide work of **Mr.D.PALANIVEL RAJAN (0720108012)** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

GUIDE

(Mrs.V.VANITHA)

HEAD OF THE DEPARTMENT

(Dr.S.THANGASAMY)

The candidate with **University Register No. 0720108012** was examined by us in Project Viva-Voce examination held on 05-05-2009

5/5/09

Internal Examiner

5/5/09

External Examiner

VELALAR COLLEGE OF ENGINEERING AND TECHNOLOGY

THINDAL, ERODE-638012

DEPARTMENT OF EEE

FIRST

NATIONAL CONFERENCE ON

"APPLICATION OF EMERGING TECHNOLOGIES IN ELECTRICAL SCIENCE"

NAETES-09

CERTIFICATE

Awarded to

D.Palanivel Rajan

M.E (Computer Science Engineering),

Kumaraguru College of Technology.

For presenting a paper titled " CONSOLIDATE AND ADVANCE: AN EFFICIENT QOS MANAGEMENT TECHNIQUE IN HETEROGENEOUS WIRELESS SENSOR NETWORKS " in the First National conference on " Application of Emerging Technologies in Electrical Science " during 09th April 2009 at Velalar college of Engineering and Technology, Erode.

COORDINATOR	CONVENOR	SESSION CHAIR	PRINCIPAL	ADMINISTRATIVE DIRECTOR
M.SATHISKUMAR	Dr.M.SRINIVASAN		Dr.K.PALANIISAMY	Dr.P.SABAPATHI

ABSTRACT

Wireless Sensor Networks has become an extensive explorative area during the last few years. It is a new class of distributed systems that are an integral part of the physical space they inhabit.

In wireless sensor network the sensor node, is also known as a 'mote', a node in a wireless sensor network that is capable of performing some processing, gathering sensory information and communicating with other connected nodes in the network.

In this the discussion for repositioning of the Consolidating and Advancing node, (CAN: Principal Node) to improve network lifetime in terms of energy. Then the QoS parameters such as latency, reliability and throughput. It address issues related to its repositioning such as the time and position of relocation and the control of its movement without causing negative impact on the performance of the WSN.

Then the Mobility factor is exploited to support QoS requirements. In this a scheme called Energy Conserving Relocation (ECR) to pursue relocation of CAN to a safe location on demand. ECR performs relocation based on the minimum energy concept of sensor nodes. The concept repositioning adds a new dimension to the existing Heterogeneous Wireless Sensor Network.

ஆய்வுச்சுருக்கம்

கடந்த சில ஆண்டுகளாக கம்பியில்லா உணரி அமைப்பு பரந்து விரிந்த ஒரு அமைப்பு இந்த அமைப்பு சிறு சிறு பகுதிகளை கொண்டது.

இந்த கம்பியில்லா வலை அமைப்பு என்பது மிக நுட்பமாண கருத்துக்களை கொணடது மற்றும் அதன் மற்ற கட்டமைப்புகளுடன் தொடர்பு கொண்டுள்ளது கட்டமைப்பின் ஆயுட்காலம் கணுக்களின் ஆற்றலை பொருத்து அமைகிறது கம்பியில்லா உணரி அமைப்பு தரமான சேவையின் முலம் தாமதம்,நம்பகத்தன்மை,மூலதனம் மூலமாக வழங்குகிறது கம்பியில்லா உணரி அமைப்பை பயன்படுத்தி இடத்தையும் நேரத்தையும் பொருத்து மாறுவதால் எதிர்மறையான விளைவுகள் ஏற்படுவதில்லை .

தரமான சேவையின் தேவைகள் நகரும் தன்மை கொண்ட கணுக்களினால் அழிக்கப்படுகிறது இடம்மாறும் ஆற்றல் சேமிக்கும் முறை மூலமாக கணுவின் இடம்மாறுதல் நடைபெறுகிறது ஆற்றல் குறைவாக உள்ள கணுவை தேர்ந்தெடுத்து அந்த கணுவின் இடத்தில் ஒருமித்த ஒருபடித்தான கணுவில் வைக்கிறோம்.கணுவின் இடம்மாறுதல் மூலமாக இருபடித்தான கம்பியில்லா உணரி அமைப்பு ஒரு பரிணாமத்தை தோற்றுவிக்கிறது

ACKNOWLEDGEMENT

I express my profound gratitude to our Chairman **Padmabhusan Arutselver Dr. N. Mahalingam B.Sc, F.I.E.**, Correspondent **Shri.Balasubramanian M.Com,M.B.A(USA)** and Joint Correspondent **Dr.A.Selvakumar Ph.D.** for giving this great opportunity to pursue this course.

I thank **Dr.Joseph V.Thanikal Ph.D., PDF., CEPIT.,** Principal, and **Prof.R.Annamalai,** Vice Principal Kumaraguru College of Technology, Coimbatore, for providing me with the necessary facilities and Infrastructure to work on this project.

I express my sincere thanks to **Dr.S.Thangasamy,Ph.D.,** Dean and HOD, Department of Computer Science and Engineering, for being my greatest of inspiration and for embedding the quest for innovative ideas.

I express my deep sense of gratitude and gratefulness to Project Coordinator as well as my guide **Mrs.V.Vanitha, M.E.,** Asst. Professor, Department of Computer Science and Engineering, for her supervision, tremendous patience, active involvement and guidance.

I would like to convey my honest thanks to all Teaching staff members and Non Teaching staff members of the department for their support. I would like to thank all my classmates who gave me proper light moments and study breaks apart from extending some technical support whenever I needed them most.

I dedicate this Project work to my **Parents** for no reason but feeling from bottom of my heart, without their love this work wouldn't possible

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
WSN	Wireless Sensor Networks
QoS	Quality of Services
CAN	Consolidate and Advancing Node

LIST OF NOTATIONS

L_n [ ]	List of neighbors
T_n	Total number of nodes
L	Set of links between nodes
I_n [ ]	Initial set of nodes;
C_d	Critical Distance
N_x, N_y	Node's X and Y Position
E_{vx}, E_{vy}	Event's X and Y position
C_x, C_y	Centroid's X and Y Position
L_{an}	List of Active Nodes
L_{on}	List of Operating Nodes
N_min, N_max	Minimum and Maximum Energy Node
E_n, E_CAN	Node Energy, CAN Energy.
E_t,	Total Energy dissipated in the transmitter of source node
E_r	Total Energy incurred in the receiver of the destination node
Tx_{ct}, Rx_{ct}, Rpx_{ct}	Transmission, Reception and Repositioning currents
E_rp	Energy costs incurred during the repositioning of CAN node
A	Area of the polygon and all of them defined in terms of x_i and y_i.

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW OF WIRELESS SENSOR NETWORKS

1.1.1. Sensor network

A wireless sensor network is a collection of nodes organized in a network. Each node consists of one or more microcontrollers, CPUs or DSP chips, a memory and a RF transceiver, a power source such as batteries and accommodates various sensors and actuators. The nodes communicate wirelessly and often self-organize after being deployed in an ad hoc fashion.

1.1.2. Components of sensor networks

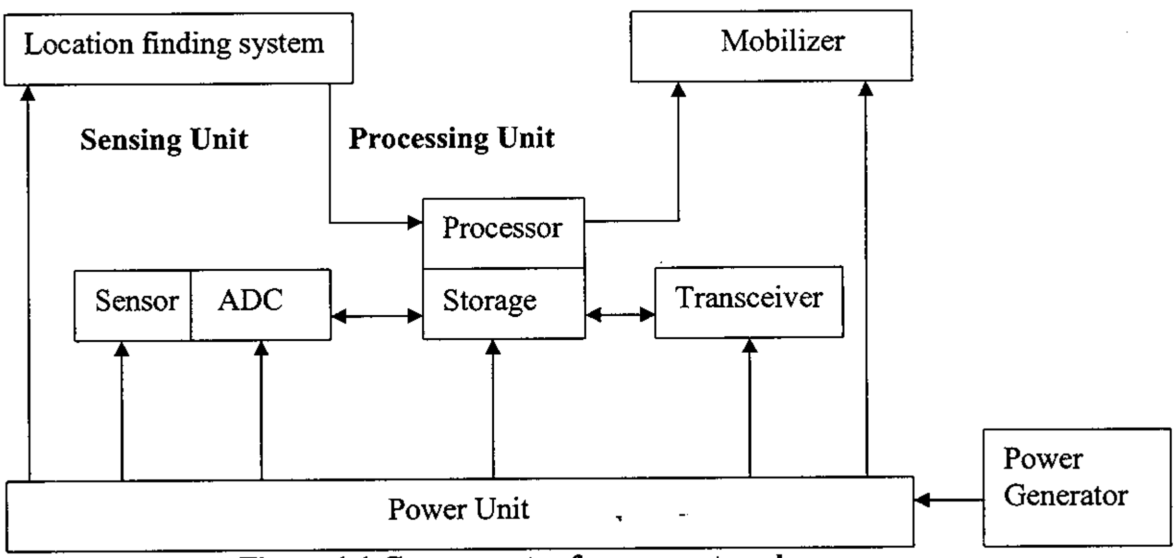

Figure 1.1.Components of sensor networks

- **Sensing unit**

Sensing units are usually composed of two subunits: sensors and analog to digital converters (ADCs).The analog signals produced by the sensors are converted to digital signals by the ADC, and then fed into the processing unit.

- **Processing Unit**

The processing unit which is generally associated with a small storage unit manages the procedures that make the sensor nodes collaborate with the other nodes to carry out the assigned sensing tasks.

- **Transceiver Unit**

A transceiver unit connects the nodes to the networks.

- **Mobilizer**

The mobilizer may be needed in some applications where the node is required to move to carry out the required tasks.

- **Location finding system**

Most of the sensors require location information for routing as the routing techniques may require knowledge of the location sensed with high accuracy

- **Power Unit**

One of the most important components of a sensor node is the power unit. Power units may be supported by a power scanning unit such as solar cells.

1.2 APPLICATIONS OF SENSOR NETWORKS

Wireless Sensor Networks have a wide range of applications such as,

1. Military applications
   - i. Monitoring friendly forces and equipment.
   - ii. Battlefield surveillance.
   - iii. Nuclear, biological and chemical attack detection.
2. Environmental Applications
   - i. Forest fire detection.
   - ii. Bio-complexity mapping of the environment.
   - iii. Flood detection.

3. Health applications

   i. Tele-monitoring of human physiological data.

   ii. Tracking and monitoring doctors and patients inside a hospital.

   iii. Drug administration in hospitals.

4. Home application

   i. Home automation.

   ii. Smart environment

In order to enable reliable and efficient observation and initiate right actions, physical phenomenon features should be reliably detected/estimated from the collective information provided by sensor nodes. Moreover, instead of sending the raw data to the nodes responsible for the fusion, sensor nodes use their processing abilities to locally carry out simple computations and transmit only the required and partially processed data. Hence, these properties of WSN impose unique challenges for development of communication protocols in such architecture. The intrinsic properties of individual sensor nodes, pose additional challenges to the communication protocols in terms of energy consumption.

1.3 CHALLENGES OF WIRELESS SENSOR NETWORKS

A sensor network design is influenced by many factors, which include,

- **Energy efficiency/system lifetime**

As sensor nodes are battery-operated, protocols must be energy-efficient to maximize system life time. System life time can be measured such as the time until half of the nodes die or by application-directed metrics, such as when the network stops providing the application with the desired information about the phenomena.

- **Fault Tolerance**

Some sensor nodes may fail or be blocked due to lack of power, have physical damage or environmental interference. The failure of sensor nodes should not affect the overall task of the sensor network. This is the reliability or fault tolerance issue. Fault tolerance is the ability to sustain sensor network functionalities without any interruption due to sensor node failures.

- **Scalability**

The number of sensor nodes deployed in studying a phenomenon may be in the order of hundreds or thousands. Depending on the application, the number may reach an extreme value of millions. The new schemes must be able to work with this number of nodes.

- **Production Costs**

Since the sensor networks consist of a large number of sensor nodes, the cost of a single node is very important to justify the overall cost of the networks. If the cost of the network is more expensive than deploying traditional sensors, then the sensor network is not cost-justified. As a result, the cost of each sensor node has to be kept low

- **Environment**

Sensor nodes are densely deployed either very close or directly inside the phenomenon to be observed. Therefore, they usually work unattended in remote geographic areas. They may be working in busy intersections, in the interior of large machinery, at the bottom of an ocean, inside a twister, on the surface of an ocean. They work under high pressure in the bottom of an ocean, in harsh environments such as debris or a battlefield, under extreme heat and cold such as in the nozzle of an aircraft engine or in arctic regions, and in an extremely noisy environment such as under intentional jamming.

- **Hardware Constraints**

A sensor node is made up of four basic components: a sensing unit, a processing unit, a transceiver unit and a power unit. Sensing units are usually composed of two subunits: sensors and analog to digital converters (ADCs). The analog signals produced by the sensors based on the observed phenomenon are converted to digital signals by the ADC, and then fed into the processing unit. The processing unit, which is generally associated with a small storage unit, manages the procedures that enable the sensor node collaborate with the other nodes to carry out the assigned sensing tasks. A transceiver unit connects the node to the network. One of the most important components of a sensor node is the power unit. Power units may be supported by a power scavenging unit such as solar cells. There are also other subunits, which are application dependent. Most of the sensor network routing techniques and sensing tasks require the knowledge of location with high accuracy.

- **Sensor Network Topology**

Sheer numbers of inaccessible and unattended sensor nodes, which are prone to frequent failures, make topology maintenance a challenging task. Hundreds to several thousands of nodes are deployed throughout the sensor field.

a) Pre-Deployment Phase

Sensor nodes can be either thrown in mass or placed one by one in the sensor field. They can be deployed by dropping from a plane, delivering in an artillery shell, rocket or missile, throwing by a catapult , placing in factory, and placing one by one either by a human or a robot. Although the sheer number of sensors and their unattended deployment usually preclude placing them according to a carefully engineered deployment plan, the schemes for initial deployment must reduce the installation cost, eliminate the need for any pre-organization and preplanning, increase the flexibility of arrangement, and promote self-organization and fault tolerance.

***b)* Post-Deployment Phase**

After deployment, topology changes are due to change in sensor nodes position, reach ability (due to jamming, noise, moving obstacles, etc.), available energy, malfunctioning, and task details. Sensor nodes may be statically deployed. However, device failure is a regular or common event due to energy depletion or destruction. It is also possible to have sensor networks with highly mobile nodes. Besides, sensor nodes and the network experience varying task dynamics, and they may be a target for deliberate jamming. Therefore, sensor network topologies are prone to frequent changes after deployment.

c) **Re-Deployment of Additional Nodes Phase**

Additional sensor nodes can be re-deployed at any time to replace the malfunctioning nodes or due to changes in task dynamics. Addition of new nodes poses a need to re-organize the network. Coping with frequent topology changes in an ad hoc network that has myriads of nodes and very stringent power consumption constraints requires special routing protocols.

- **Transmission Media**

In a multi-hop sensor network, communicating nodes are linked by a wireless medium. These links can be formed by radio, infrared or optical media. To enable global operation of these networks, the chosen transmission medium must be available worldwide. One option for radio links is the use of *Industrial, Scientific and Medical* (ISM) bands, which offer license free communication in most countries.

- **Power Consumption**

The wireless sensor node, being a microelectronic device, can only be equipped with a limited power source. In some application scenarios, replenishment of power resources might be impossible. Sensor node lifetime, therefore, shows a strong dependence on battery lifetime. The malfunctioning of few nodes can cause significant topological changes and might require rerouting of packets and re-organization of the network. In sensor networks, power efficiency is an important performance metric, directly influencing the network lifetime. Application specific protocols can be designed by appropriately trading off other performance metrics such as delay and throughput with power efficiency. The main task of a sensor node in a sensor field is to detect events, perform quick local data processing, and then transmit the data. Power consumption can hence be divided into three domains: *sensing*, *communication*, and *data processing.*

CHAPTER 2

LITERATURE REVIEW

2.1 QoS Support in Traditional Data Networks

Supporting QoS in wired networks can generally be obtained via the over-provisioning of resources and/or traffic engineering. With the method of over-provisioning, it provide satisfactory services to bandwidth-hungry multimedia applications the abundant resources can be added in the network. In the method based on traffic engineering, we classify our users/applications in service classes and assign each class a different priority, there are two approaches based on traffic engineering are exploited to achieve QoS, reservation-based and reservation-less approaches.

In the reservation-based approach, network resources are assigned according to an application's QoS request and subject to bandwidth management policy. This is employed in Asynchronous Transfer Mode (ATM) and also the approach of the InterServ model in the Internet. In the reservation-less approach, no reservation is required. QoS is achieved via some strategies such as admission control, policy managers, traffic classes, and queuing mechanisms. Admission control strategy decides if a node can access the network and guarantees that once the node obtains the permission, it will be served with the QoS it is requesting. Policy managers ensure that no node will violate the type of services it is pre-assigned. Traffic classes differentiate the priority of data packets and they thereby achieve a particular per-hop behavior at each intermediate node. Queuing mechanisms are responsible for dropping the packets with lower priority in the case of congestion. This approach is well known as the approach of the DiffServ model in the Internet.

Infrastructure-based wireless networks, such as Wireless Local Area Networks (WLANs) and Broadband Wireless Access Networks (BWANs), are the extension of wired networks, so that the connections can be extended to mobile users. All mobile hosts in a communication cell can reach a base station in one hop. QoS challenges in this context mainly arise from the scarce bandwidth and the complexity of user mobility during the last wireless hop. Thus, it is intuitive for us to integrate the QoS architecture deployed in wired networks with wireless MAC protocols. Wireless MAC protocols may provide data traffic of differentiated classes with corresponding access priorities over the shared wireless medium so that the overall QoS can be supported. Unfortunately, QoS mechanisms used to support QoS in wired data networks cannot be directly applied to ad hoc networks because of the bandwidth constraint and dynamic network topology. In this context, we are required to implement complex QoS functionality with limited available resources in a highly dynamic environment.

In the literature, QoS Support in ad hoc networks includes QoS model, QoS resource reservation signaling, QoS routing, and QoS Medium Access Control (MAC). A QoS model specifies architecture and impacts the functionality of other QoS components. For instance, if the network is only required to provide differentiated services, signaling for every flow state is unnecessary. QoS signaling, the functionality of which is determined by the QoS model, acts as a control center in the QoS support system. It coordinates the behavior of QoS routing, QoS MAC, and other components.

2.2 QoS Requirements in WSN

A generic wireless sensor network is composed of a large number of sensor nodes scattered in a terrain of interest. Each of them has the capability of collecting data about an ambient condition, like temperature, pressure, humidity, noise, lighting condition and sending data reports to a sink node. Since there exist many envisioned applications in WSNs and their QoS requirements may be very different, it is impossible for us to analyze them individually. Also, it is unlikely that there will be a "one-size-fits-all" QoS support solution for each application. However, the QoS requirements imposed by the applications on the network, we can initially separate QoS requirements using other perspectives from the networking perspective.

In applications involving event detection and target tracking, the failure to detect or extracting wrong or incorrect information regarding a physical event may arise from many reasons. It may be due to the deployment and network management, the location where the event occurs may not be covered by any active sensors. Intuitively, we can define coverage or the number of active sensors as parameters to measure the QoS in WSNs. In addition, the above failure may be caused by the limited functionality of sensors, inadequate observation accuracy or the low reporting rate of sensors. They can define observation accuracy or measurement errors as parameters to measure QoS. Further, it may be induced by information loss during the delivery. This can also define some information transportation related parameters to measure QoS. However, this separation of QoS perspectives is not absolute since a common application requirement such as the performance measure associated with event detection may involve all of them. That can be described in two perspectives of QoS in WSNs.

2.2.1. Application-specific QoS: - In this perspective, we may consider QoS parameters such as coverage, exposure, measurement errors, and optimum number of active sensors. The applications impose specific requirements on the deployment of sensors, the number of active sensors, and the measurement precision of sensors and so on, which are directly related to the quality of applications.

2.2.2. Network QoS: - In this perspective, we consider how the underlying communication network can deliver the QoS-constrained sensor data while efficiently utilizing network resources. Although we cannot analyze each possible application in WSNs, it is sufficient for us to analyze each class of applications classified by data delivery models, since most applications in each class have common requirements on the network. From the point of view of network QoS, we are not concerned with the application that is actually carried out, we are concerned with how the data is delivered to the sink and corresponding requirements. Generally, there are three basic data delivery models, they are event-driven, query-driven, and continuous delivery models. Before presenting the application requirements, we would like to provide some factors that characterize them as follows:

(i) End-to-end: The application may require end-to-end or non-end-to-end performance

(ii) Interactivity: The application may be interactive or non interactive

(iii) Characteristics: The application may or may not be delay tolerant

(iv) Criticality: The application may or may not be mission critical

i) Event-driven

Most event-driven applications in WSNs are interactive, delay intolerant (real-time), mission critical and non-end-to-end applications. It means that the events sensors are expected to observe are very important to the success of the application. The application needs to detect these events and accordingly takes an appropriate action as quickly as possible and as reliably as possible.

There are several important points should be mentioned. First, the application itself is not end-to-end, the one end of the application is the sink, the other end is not a single sensor node, but a group of sensor nodes within the area that is influenced by the event. Second, the data flows from these sensors are likely to be highly correlated and thereby containing much redundancy. Third, the data traffic generated by a single sensor may be of very low intensity. However very bursty traffic may be generated by a set of sensors due to a common event or a phenomenon known as event showers.

Finally actions in response to the detected event may need to be distributed to sensors or actuators as quickly as possible and as reliably as possible. These sensors and actuators may not be the same set of sensors that notified the sink about this event. This data delivery model involves many typical WSN applications that require event detection and signal estimation/tracking, sensing of and response to an emergency due to chemical release in a building

ii) Query-driven

Most query-driven applications in WSNs are interactive, query-specific delay tolerant, mission critical and non-end-to-end applications. To save energy, queries can be sent on demand. This data delivery model is similar to the event-driven model except that the data is pulled by the sink while the data is pushed to the sink in the event driven model. The applications still need to receive these desired data as quickly as possible and as reliably as possible. The important points mentioned for event-driven delivery are also relevant for query-driven delivery. The query may also be used to manage and reconfigure the sensor nodes. For example, if the sink wants to upgrade the software on the sensor nodes, reconfigure the sending rate, or change the sensor mission, the sink can send out a command to execute these changes. It should be noted that the commands from the sink constitute one-way traffic and require high reliability.

iii) Continuous

In the continuous model, sensors send their data continuously to the sink at a pre-specified rate. Real-time voice, image, or video data: Real-time data is delay-constrained and has a certain bandwidth requirement. Packet losses can be tolerated to a certain extent. As such, they are not end-to-end applications. Non-real-time data: The sink may want to collect periodic data from the sensor field. In this context, delay and packet losses are both tolerated.

iv) Hybrid models

In many applications, the data delivery models described above may coexist in the network. Thus, it may require a mechanism to accommodate different types of QoS-constrained traffic. These requirements are summarized in Table I, and more importantly, we note that there are some differences in application requirements between WSNs and traditional networks. First of all, applications in WSNs are no longer end-to-end applications. Second, bandwidth is not the main concern for a single sensor node. However, bandwidth may be an important concern for a group of sensors for certain time periods due to the bursty nature of sensor traffic. Third, packet losses in traffic generated by one single sensor node can be tolerated to a certain extent since there always exists much redundancy in the data. Finally, most applications in WSNs are mission critical, which reflects the importance of applications. In QoS parameters to measure the QoS support in WSNs. We thereby need to propose some new non-end-to end QoS parameters. As a whole, we term such non-end-to-end
parameters collective QoS parameters. These are:

(i) Collective latency

(ii) Collective packet loss

(iii) Collective bandwidth

(iv) Information throughput

In this a collective parameter to characterize jitter since multimedia applications are not major applications of WSNs. Besides, we do not provide a precise definition of each collective QoS parameter. Instead, we utilize an example to demonstrate the novel concept of collective QoS parameters. For instance, in an event-driven wireless sensor network. The sensors residing within a certain radius of the event are reporting the information about this event to the sink. In this context, collective latency is defined as the difference between the time at which the first packet related to this event is generated by the source sensors and the time at which the last packet related to this event or the last packet used to make a decision arrives at the sink. Collective packet loss is defined as the number of packets related to this event lost during information delivery. Collective bandwidth is defined as the bandwidth that the reporting of the event requires. To sum up, the sink should be concerned about an end-to-end event, instead of the packets from individual sensors. In addition, it should consider information throughput at the sink from a set of for individual sensors.

2.3 Design Issues in WSN

Depending on the application, different architectures and design goals/constraints have been considered for sensor networks. Since the performance of a routing and MAC protocols are closely related to the architectural model, in this section we strive to capture architectural design issues and highlight their implications. Later we will analyze the complexity of supporting QoS traffic in light of these design variations.

i) Network Dynamics There are three main components in a sensor network. These are the sensor nodes, sink and monitored events. Aside from the very few setups that utilize mobile sensors, most of the network architectures assume that sensor nodes are stationary. On the other hand, supporting the mobility of sinks or cluster-heads (gateways) is sometimes deemed necessary. Routing messages from or to moving nodes is more challenging since route stability becomes an important optimization factor, in addition to energy, bandwidth etc. The sensed event can be either dynamic or static depending on the application. For instance, in a target detection/tracking application, the event (phenomenon) is dynamic whereas forest monitoring for early fire prevention is an example of static events. Monitoring static events allows the network to work in a reactive mode, simply generating traffic when reporting. Dynamic events in most applications require periodic reporting and consequently generate significant traffic to be routed to the sink.

ii) Node Deployment Another consideration is the topological deployment of nodes. This is application dependent and affects the performance of the routing protocol. The deployment is either deterministic or self-organizing. In deterministic situations, the sensors are manually placed and data is routed through pre-determined paths. In addition, collision among the transmissions of the different nodes can be minimized through the pre-scheduling of medium access. However in self-organizing systems, the sensor nodes are scattered randomly creating an infrastructure in an ad hoc manner. In that infrastructure, the position of the sink or the cluster-head is also crucial in terms of energy efficiency and performance. When the distribution of nodes is not uniform, optimal clustering becomes a pressing issue to enable energy efficient network operation.

iii) Node Communications During the creation of an infrastructure, the process of setting up the routes is greatly influenced by energy considerations. Since the transmission power of a wireless radio is proportional to distance squared or even higher order in the presence of obstacles, multi-hop routing will consume less energy than direct communication. However, multi-hop routing introduces significant overhead for topology management and medium access control. Direct routing would perform well enough if all the nodes were very close to the sink. In Most of the time sensors are scattered randomly over an area of interest and multi-hop routing becomes unavoidable. Arbitrating medium access in this case becomes cumbersome.

iv) Data Delivery Models Depending on the application of the sensor network, the data delivery model to the sink can be continuous, event-driven, query-driven and hybrid. In the continuous delivery model, each sensor sends data periodically. In event-driven and query-driven models, the transmission of data is triggered when an event occurs or a query is generated by the sink. Some networks apply a hybrid model using a combination of continuous, event-driven and query-driven data delivery. The routing and MAC protocols are highly influenced by the data delivery model, especially with regard to the minimization of energy consumption and route stability. In a habitat monitoring application where data is continuously transmitted to the sink, a hierarchical routing protocol is the most efficient alternative. This is due to the fact that such an application generates significant redundant data that can be aggregated on route to the sink, thus reducing traffic and saving energy. In addition, in continuous data delivery model time-based medium access can achieve significant energy saving since it will enable turning off sensors' radio receivers. CSMA medium access arbitration is a good fit for event-based data delivery models since the data is generated sporadically.

v) Node Capabilities In a sensor network, different functionalities can be associated with the sensor nodes. In early work on sensor networks, all sensor nodes are assumed to be homogenous, having equal capacity in terms of computation, communication and power. However, depending on the application a node can be dedicated to a particular special function such as relaying, sensing and aggregation since engaging the three functionalities at the same time on a node might quickly drain the energy of that node. Some of the hierarchical protocols proposed in the literature designate a cluster-head different from the normal sensors. While some networks have picked cluster-heads from the deployed sensors in other applications a cluster-head is more powerful than the sensor nodes in terms of energy, bandwidth and memory. In such cases, the burden of transmission to the sink and aggregation is handled by the cluster-head.

iv) Data Aggregation/Fusion Since sensor nodes might generate significant redundant data, in some applications similar packets from multiple nodes can be aggregated so that the number of transmissions would be reduced. Data aggregation is the combination of data from different sources by using functions such as suppression (eliminating duplicates), min, max and average. Some of these functions can be performed either partially or fully in each sensor node, by allowing sensor nodes to conduct in-network data reduction. Recognizing that computation would be less energy consuming than communication, substantial energy savings can be obtained through data aggregation. This technique has been used to achieve energy efficiency and traffic optimization in a number of routing protocols. In some network architectures, all aggregation functions are assigned to more powerful and specialized nodes. Data aggregation is also feasible through signal processing techniques. In that case, it is referred as data fusion where a node is capable of producing a more accurate signal by reducing the noise and using some techniques such as beam forming to combine the signals. Data aggregation makes medium access control complex since redundant packets will be eliminated and such elimination will require instantaneous medium access arbitration. In such case, only CSMA and CDMA-based MAC protocols are typically applicable leading to an increase in energy consumption.

2.4 QoS Requirements for WSN in Layers

The definition of QoS requirements for WSN Layers can be .given as follows.

2.4.1 Application Layer The QoS requirements in the application layer are specified by users. The following QoS requirements for WSNs applications are: System Lifetime, Response Time, Data novelty, Detection Probability Data Reliability and Data Resolution. WSNs are often required to sustain its functionalities for a certain time period. System Lifetime is defined as the time from system deployment up to the time when it cannot satisfy users requirements. In on-demand WSNs applications, Response Time refers to the latency from the time that a user sends a query to the time that the user receives the response. Many applications, such as battlefield monitoring, require users to receive data in a timely manner. Data novelty refers to the latency from the time an event is detected by a sensor to the time the data about the event arrive at storage sensors or sink points. In addition to data transmission,

Detection Probability refers to the probability that a real world phenomenon can be detected and reported to a user. Two requirements on data quality, data reliability and data Resolution refer to the degree that the reported data corresponds to real world phenomena and the sampling rate in the spatial/temporal scale, respectively. Data reliability describes the accuracy of the data and Data Resolution imposes temporal/spatial granularity on the data.

2.4.2 Transport Layer In this QoS requirement for the transport layer: Reliability, Bandwidth, Latency and Cost. In this the concept of packets and unique packets are differentiated as defined by the "collective" concept introduced in from a sensor's point of view, unique packets refer to the packets containing data that are not correlated with the already received data. All of the QoS requirements within the transport layer use the collective concept, which means only unique packets are counted as received by the destination. Reliability refers to the percentage of unique packets successfully received from all sending sources in reference to those that were actually transmitted. Bandwidth refers to the number of unique packets received per unit time from all sending sources. Latency refers to the shortest total delay at the intermediate nodes/channels in transmitting a unique packet from all sending sources to a destination, which includes propagation delay, queuing delay and routing delay. Cost is defined as the number of transmissions to retrieve a unique packet from all sending sources.

2.4.3 Network Layer In this QoS requirement for the network layer: Path Latency, Routing Maintenance, Congestion Probability, Routing Robustness and Energy Efficiency. Path Latency refers to the average number of hops between all source destination pairs in the network. Routing Maintenance refers to the energy consumption rate to maintain routes between all source destination pairs. Congestion Probability is the probability that the traffic load on any path exceeds the bottleneck capacity of all the links on the path. Routing Robustness is defined as the maximal probability of packet loss that routing must sustain. Energy Efficiency measures the amount of energy consumed to transmit a data packet along a path.

2.4.4 Connectivity Maintenance Layer In this QoS requirement for the connectivity maintenance layer: Network Diameter, Network Capacity, Average Path Cost, Connectivity Robustness, and Connectivity Maintenance. Network Diameter is defined as the maximal transmission latency between two sensors in the formed network topology. Network Capacity is defined as the number of packets that can be transmitted concurrently in the network. Average Path Cost is defined as the average amount of energy consumed to transmit one packet between all source destination pairs. Connectivity Robustness is defined as the maximal allowed number of failed sensors/links that the network connectivity must sustain. Connectivity Maintenance measures the energy consumption rate to maintain a connected network topology.

2.4.5 Coverage Maintenance Layer In this the QoS requirements for the coverage maintenance layer: Coverage Percentage, Coverage Reliability, Coverage Robustness and Coverage Maintenance. Coverage Percentage refers to the percentage of area monitored by at least one sensor. Coverage Reliability refers to the minimal allowed sensing probability. Coverage Robustness is defined as the minimal number of sensors monitoring the same location. Coverage Maintenance measures the number of messages exchanged to provide and maintain network coverage.

2.4.6 MAC Layer In this the QoS requirements for the MAC layer Communication Range, Throughput, Transmission Reliability and Energy Efficiency. Communication Range refers to the maximal distance of one-hop data transmission. Throughput refers to the maximal number of data frames that can be transmitted successfully by the MAC layer per unit time. Transmission Reliability refers to the percentage of successfully transmitted frames. Energy Efficiency measures the amount of energy consumed to successfully transmit one frame within one-hop.

2.4.7 Physical Layer The physical layer describes wireless sensor capabilities, which encompass wireless unit capabilities, processor capabilities and sensing unit capabilities. Specifically, wireless unit capabilities refer to Channel Speed, Coding and RF Power. Processor capabilities are Location capabilities, Timing capabilities, Processing Speed and Computation Power. Sensing unit capabilities include Measurement Accuracy, Sensing Range and Sensing Power. A sensor's physical capabilities impose resource constraints on the QoS requirements of other layers. For wireless unit capabilities, Channel Speed impacts Throughput in the MAC layer; Coding impacts Throughput and Transmission Reliability in the MAC layer; RF Power impacts Communication Range, Transmission Reliability and Energy Efficiency in the MAC layer. For processor capabilities, Location and Timing capabilities impact Location Accuracy and Timing Accuracy respectively; Processing Speed determines Processing Latency in the data management layer; Computation Power impacts Computation Cost, Data Abstraction and Data Accuracy in the data management layer, as well as Energy Consumption in the location/time service layer. For sensing unit capabilities, Measurement Accuracy impacts Coverage Reliability and Coverage Robustness in the coverage maintenance layer; Sensing Range affects Coverage Percentage in the coverage maintenance layer; Sensing Power affects all the requirements in the coverage maintenance layer.

2.5 Limitations for QoS Support in WSN

Wireless sensor networks can be used for many missions critical applications such as target tracking in battlefields, and habitat monitoring in forests. In these applications, reliable and timely delivery of sensory data plays a crucial role in the success of the mission. Specifically, the above-mentioned sensor network applications share the following characteristics:

1) Resource constraints: The constraints on resources involve energy, bandwidth, memory, buffer size, processing capability, and limited transmission power. Among them, energy is a primary concern since energy is severely constrained at sensor nodes and it may not be feasible to replace or recharge the battery for sensor nodes that are often expected to work in a remote or inhospitable environment. These constraints impose an essential requirement on any QoS support mechanisms in WSNs. Computation intensive algorithms, expensive signaling protocols, or overwhelming network states maintained at sensors are not feasible.

2) Unbalanced mixture traffic: In most applications of WSNs' application, traffic mainly flows from a large number of sensor nodes to a small subset of sink nodes with mixture of periodic and un-periodic data. Some sensory data are created un-periodically by detection of critical events at unpredictable points in time. In addition, there are other types of sensory data for periodic monitoring of environmental status. In short, sensor network applications have a mixture of periodic and un-periodic traffic types, so QoS mechanisms should be designed for an unbalanced mixture QoS-constrained traffic

3) Data redundancy: WSNs are characterized by high redundancy in the sensor data. However, while the redundancy in the data does help loosen the reliability/robustness requirement of data delivery, it unnecessarily spends much precious energy. Data fusion or data aggregation is a solution to maintain robustness while decreasing redundancy in the data, but this mechanism also introduces latency and complicates QoS design in WSNs.

4) Network dynamics: Network dynamics may arise from dynamic topology changes and unreliable nature of wireless network. Dynamic topology changes due to node mobility, failure, and addition. And unreliable nature of wireless links in large scale with thousands of densely placed nodes is hard to predict the incident of links failure. Such a highly dynamic network greatly increases the complexity of QoS support.

5) Energy balance: Energy-efficiency is a key concern in WSNs. The large number of sensor nodes involved in such networks and the need to operate over a long period of time require careful management of the energy resources. In addition, wireless communication is a major source of power consumption. Since a significant portion of the communication in WSNs is due to data gathering, it is crucial to design energy-efficient communication strategies in implementing such as routing and management. In a word, to achieve a long-lived network, energy load must be evenly distributed among all sensor nodes so that the energy at a single sensor node or a small set of sensor nodes will not be drained out very soon. And QoS support should take this factor into account. Thus the QoS support for the network may have to take at least a few of the challenges described above into account when an application is specified.

2.6 Issues for QoS Monitoring

Wireless sensor networks emerge as next generation computing platform. More and more wireless sensor network applications are need to known the whole QoS parameters in WSNs. Network monitoring in WSNs is an increasingly important, yet difficult and demanding task on WSNs' infrastructures. At present traffic is processed as quickly as possible with energy constraints, but there is no guarantee as to timeliness or actual delivery. Consequently, important business applications packets such as real-time monitoring could be lost or to be delivered too late because of displacement by less important packets. This leads to the need for QoS in order to provide a consistent predictable data delivery service for different kinds of packets. QoS is an umbrella term for a collection of technologies that allow WSNs' network-aware applications to request and receive predictable service levels in terms of QoS requirements. There is no research on QoS monitoring in WSN. In traditional network QoS distribution monitoring are capable of detecting possible QoS degradation. Clearly, if the QoS seen by the flow receiver is degraded, it is impossible for end-to-end QoS monitoring to locate the degradation. In contrast, the QoS distribution monitoring can provide more information for locating the degradation. However, QoS distribution monitoring imposes the many challenges in WSN.

1) Locating relevant monitors: In order to monitor the QoS distribution of a real-time flow, traffic information of the flow needs to be collected from relevant monitors that are metering the flow. Since different real-time flows may cross different network segments, the monitors involved in monitoring different flows are different. Thus, a QoS distribution monitoring approach needs to provide mechanisms for the monitoring application to locate relevant monitors of different flows. In addition, since the monitoring application may be mobile, such as Web-based applications, the mechanisms provided should enable the mobile monitoring application to locate relevant monitors.

2) Synchronizing the retrieval of traffic information: The synchronization of traffic information retrieval among relevant monitors is another challenge in providing QoS distribution monitoring, this includes traffic information recording, reporting and consolidation. To derive QoS-related parameters such as loss ratio, traffic information needs to be collected from relevant monitors. This is similar to performance monitoring in-distributed systems and also has the so called 'synchronization' problem in retrieving information from distributed measurement. As all physical characteristics add to the complexity of require QoS monitoring in WSNs application. So many constraints such as resource limitations, unbalanced mixture traffic, data redundancy, network dynamics, and energy balance should be considered by us. For example: Energy efficiency is a key concern in WSNs. The large number of sensor nodes involved in such networks and the need to operate over a long period of time require careful management of the energy resources.

CHAPTER 3

PROPOSED METHODOLOGY

3.1 BACKGROUND AND MODEL

3.1.1 Radio Model

The radio model is described in terms of transmit, receive and repositioning energy costs for the transfer of data message between two nodes, separated by some distance.

$$E_t=Tx_{ct}*V_t*2; \quad E_r=Rx_{ct}*V_t*2; \quad E_rp=Rpx_{ct}*V_t*2$$

where E_t and E_r denote the total energy dissipated in the transmitter of source node and that incurred in the receiver of the destination node. E_rp rep resents the energy costs incurred during the repositioning of CAN node. The parameters Tx_{ct}, Rx_{ct} and Rpx_{ct} are the transmission, reception and repositioning currents.

3.1.2 Position of Event

The nodes being deployed in a heterogeneous wireless sensor network can sometimes be threatened by events. An event is described as any unwanted occurrence which has hazardous outcomes. The position of an event is described by means of centroid equation of a polygon.

$$C_x=1/6A\sum_{i=0}^{n-1} (x_i+x_{i+1})(x_iy_{i+1}-x_{i+1}y_i)$$

$$C_y=1/6A\sum_{i=0}^{n-1} (y_i+y_{i+1})(x_iy_{i+1}-x_{i+1}y_i)$$

$$A=1/2\sum_{i=0}^{n-1} (x_iy_{i+1}-x_{i+1}y_i)$$

C_x and C_y denote the x and y coordinates of the centroid and A denotes the area of the polygon, all of them defined in terms of x_i and y_i.

3.2 PROBLEM DEFINITION

A uniformly deployed sensor field of node density 'n' is considered, where the nodes are of two types, ordinary and a CAN having differential energy. A heterogeneous sensor network is modeled by means of an edge-weighted graph G_{hwsn} (N CAN, E), with node set N {no, n1, n_i, n_n}. E denotes the energies of the sensor nodes. The objectives are (i) to improve the QoS parameters of a heterogeneous wireless sensor network, taking safety of CAN into consideration. (ii) Advance the principal node to a safe and secure location.

Assumptions Made

The following assumptions are made during the simulation

1. The nodes in the sensor field are so deployed that the neighbours of any Sensor node has in turn equidistant neighbours.
2. All nodes have similar capabilities and equal significance.
3. Each of the nodes in the battery operated and the battery is not recharged.
4. Nodes are left unattended after deployment.
5. The sensor nodes are equipped with a Global Positioning System (GPS).

3.3 ALGORITHM

The algorithm to reposition the principal node and to obtain a reasonably good QoS comprises of taking care to see that the event does not adversely affect the performance of the WSN. The event is said to occur when atleast 25% of the nodes are involved in data transmissions. The centroid computed using these nodes gives the position of the event. The position so determined implies whether the event has an impact on the nodes or not. If the event has a disastrous effect on the nodes, it is termed as an influencing event, else termed as a non-influencing event.

3.3.1 Procedure to Handle Multihop Routing to reach CAN node

This function is implemented to handle multihop routing whenever CAN is away from any node. This node relays the packet to the neighbor node with maximum energy. This process continues until the packet reaches the HEAD, contributing to the energy efficiency of the protocol.

```
Advance (N, CAN, Tn, L, En, Ecan)
 begin
  if(CAN is multihop away from a node) then
   begin
    repeat
        Ln = find_neighbour (node)
        N_max = max (energy (Ln[]))
        send(msg,node,N_max)
     until CAN is reached
   end
 end
```

TABLE 3.1

3.3.2 Procedure to Handle the Influencing Event

If the event is within the sensing radius of some nodes, then the node with minimum energy among those nodes is chosen and CAN is relocated to a position depending on whether that node is safe or in danger. A node is said to be in danger if it is within the critical distance from the event, else considered safe.

```
IE(N,CAN,Tn,L,Lan,En, Ecan)
begin
   for each node in Lan []
        Advance(N,CAN,Tn,L,En, Ecan)
   end for
  send(aggregated_msg, CAN,BaseStation)
  N_min= find_min_energy(Lan[])
  d=dist(N_min, Event)
  if (d > Cd) then
       begin
          N_min is safe
          CAN(x,y) = N_min(x+6,y+6)
       end
     else
       begin
          node is in danger
```

```
            SP(X,y) = safe_pos(nx, ny, Evx, Evy, Cd)
            CAN(x,y) SP(X,Y)
          end
      LOn[]= Lon []-N_min
      send(msg,Lon[] ,CAN)
      send(aggregated_msg, CAN,BaseStation)
    end.
```

TABLE 3.2

3.3.3 Procedure to Handle the Non-Influencing Event

If the event is not within the sensing radius of nodes, then the node with minimum energy is selected from the region of heavy traffic to decide the new position of CAN.

```
NIE(N,CAN,Tn,L,Lan,En, Ecan)
begin
   N = find_random(In[])
   Ln[ = find_neighbour(N)
   for each node in Lan[]
    Advance(N,CAN,Tn,L,En, Ecan)
  end for
   send(aggregated_msg,CAN,Basesatation)
   N_min = find_min_energy(Ln[ ])
   CAN(X,Y) = N_min(x+6,y+6}
   Lon[]= Lon -N_min
   send(msg,Lon [],CAN)
   send(aggregated_msg,CAN,Basesatation
end.
```

TABLE 3.3

3.3.4 Procedure for Consolidating and Advancing Technique:

This algorithm aims to relocate the CAN to the safety place during the occurrence of any event.

```
CAN_ECR(N, CAN, Tn, L, En, , Ecan)
begin
 repeat
  In[ ] = 1/4p [Tn]
  if (1/4th of the total nodes sense) then
begin
  C(x,y) = centroid(ln[])
  if (Ev is within the nodes sensing radius) then
    begin
     Lan[] = add node to Lan[]
     IE(N,CAN,Tn,L,Lan, En, , Ecan)
```

```
        end
      else
    NIE(N,CAN,T_n,L,L_an, E_n, E_can)
   end
  untilNetworkLifetime.
End
```

TABLE 3.4

CHAPTER 4

RESULTS AND DISCUSSIONS

The Energy efficient, QoS for sensor networks are implemented and the performances are compared based on four parameters as i) Network Lifetime, ii) Throughput iii) Latency iv) Energy.

The experiments are performed on a Pentium 2.4 GHz with 128MB of RAM on Linux using the software Ns-2.33.

SIMULATION PROPERTIES

Type	Parameters	Value
Network	Network Grid	From (0,0) to (2000,2000)
	No. of nodes	42
	Initial Energy	100J
	Data packet size	521 bytes

TABLE 4.1

In this a scheme called Energy Conserving Relocation (ECR) to pursue relocation of CAN to a safe location on demand. ECR performs relocation based on the minimum energy concept of sensor nodes.

GRAPH RESULTS:

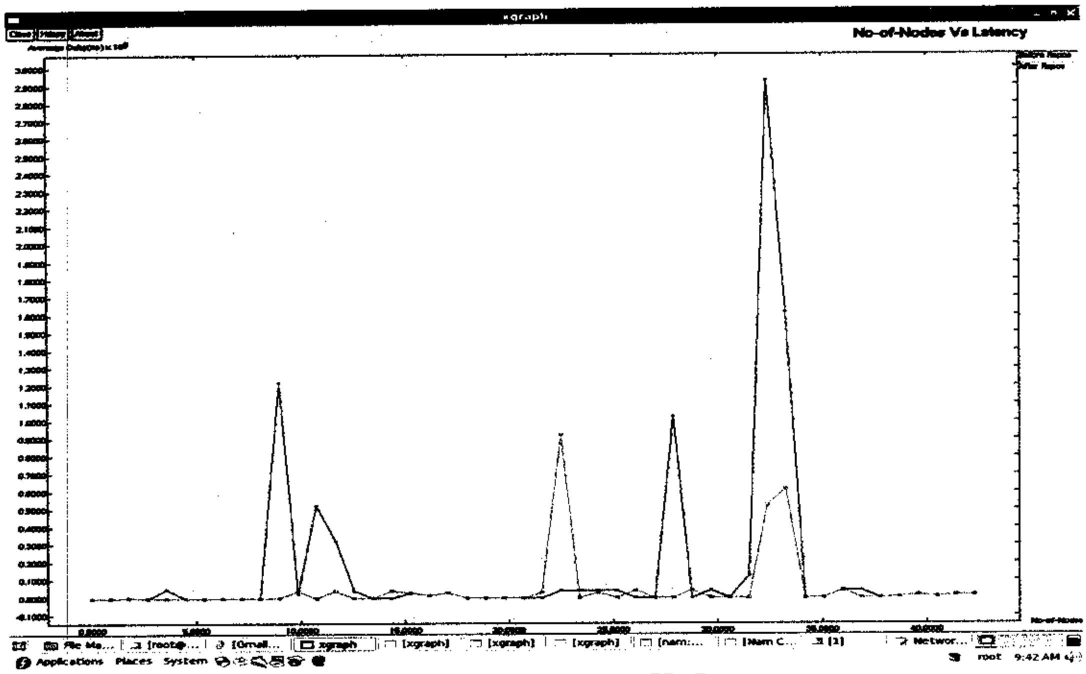

Fig 4.1 No.Of Nodes Vs Latency

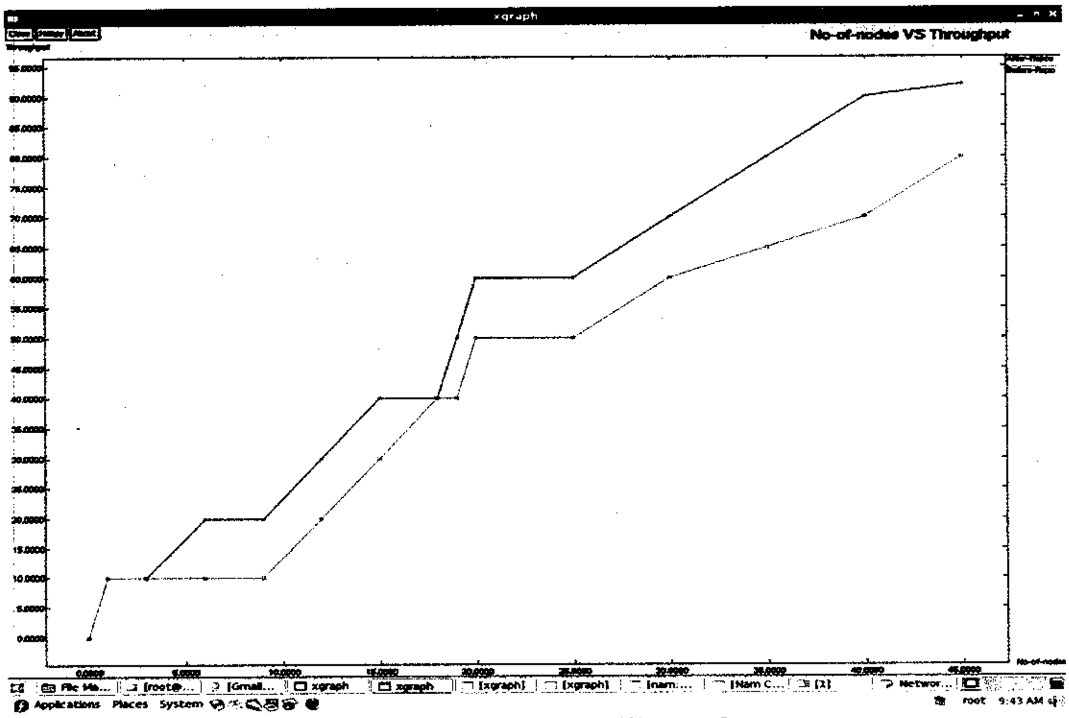

Fig 4.2 No.Of Nodes Vs Throughput

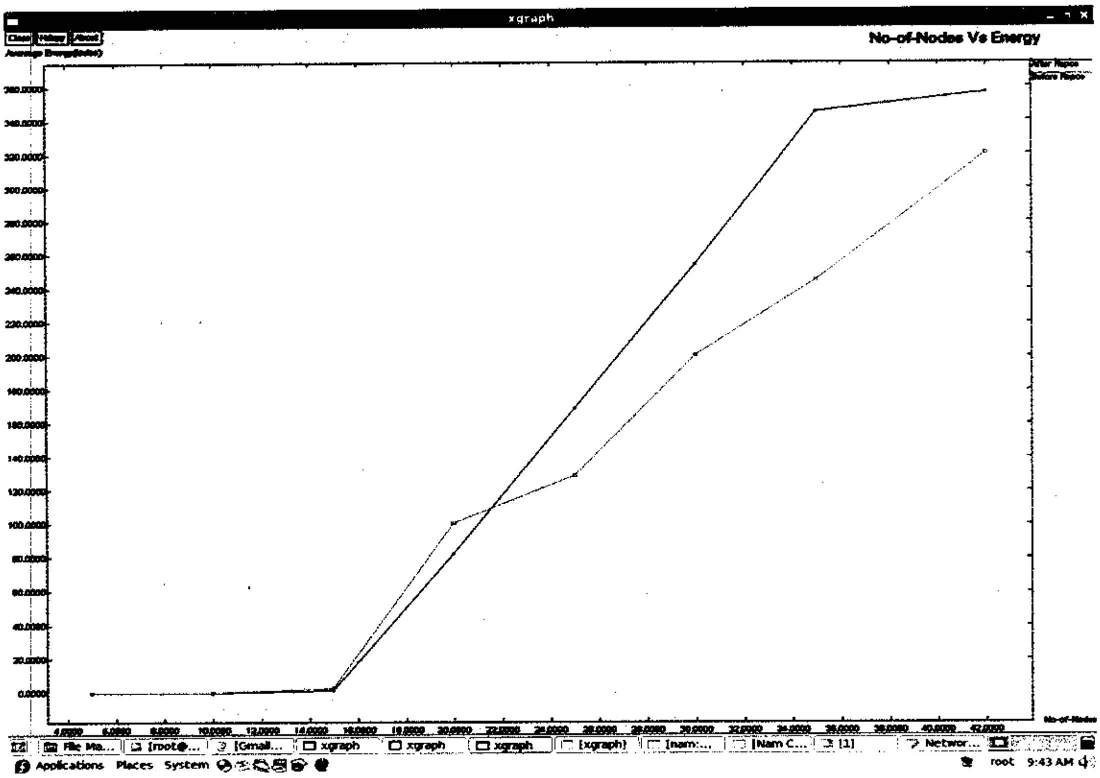

Fig 4.3 No.of Nodes Vs Energy

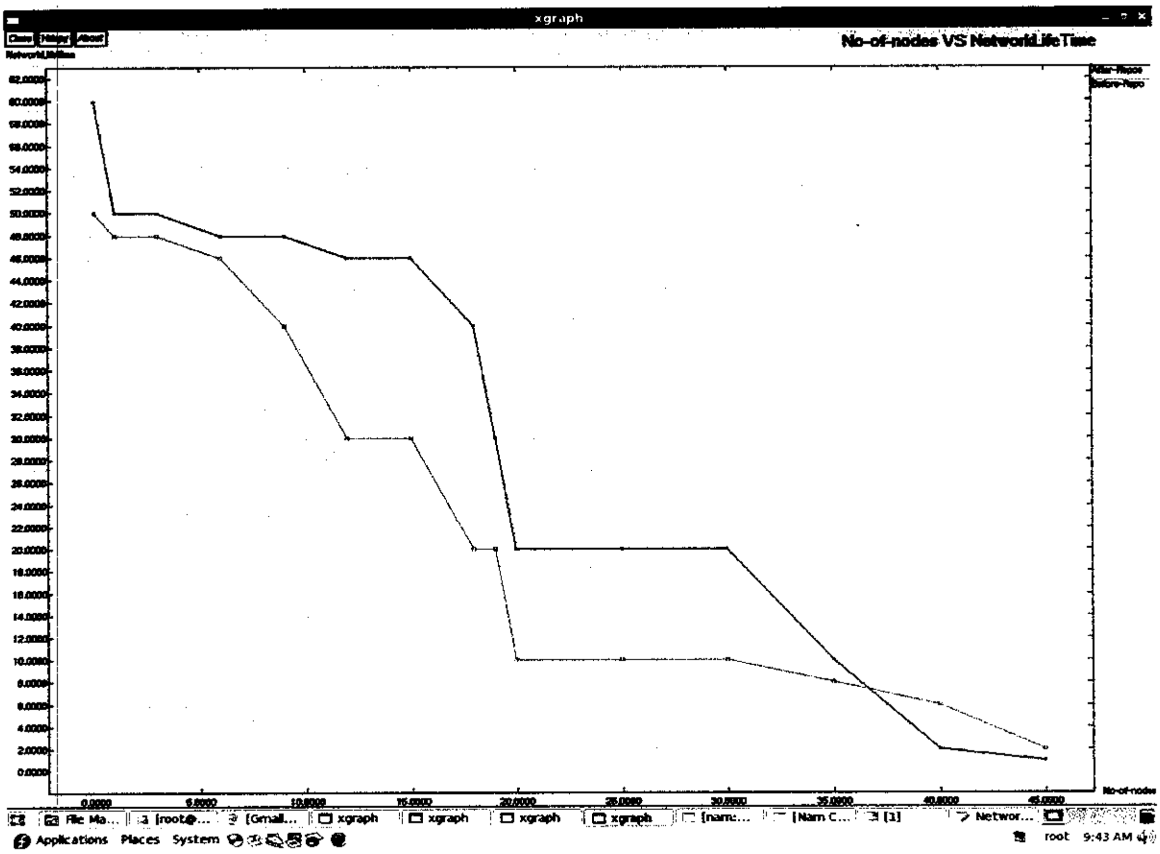

Fig 4.4 No of Nodes Vs Network Life Time

CHAPTER 5

CONCLUSION AND FUTURE OUTLOOK

5.1 CONCLUSION:

WSN is gaining a lot of importance in the real world because of its applications. We have proposed the concept of a mobile, high energy node whose function is to consolidate and advance the packets to the base station. By virtue of its mobility, moved to a suitable position to increase network lifetime leading to an overall enhancement of the QoS parameters. If a hazardous event occurs, it may be damaged and hence needs to be relocated to a better and safer position. Thus we emphasize on CAN's safety and propose a scheme called ECR which takes care of relocating CAN to a safer position. Our implementation makes best utilization of resources, avoids data redundancy, conserves energy, satisfies various QoS parameters like latency, throughput and reliability and at the same time enhances the network lifetime.

5.2 FUTURE OUTLOOK

The safety of other nodes in addition to CAN's safety and also the improve the lifetime of the sensor node and make the Reliable transmission should be taken between the nodes in the network coverage area

APPENDIX 1

Coding for Efficient QoS Technique

```
set MESSAGE_PORT 42
set BROADCAST_ADDR -1
#===========================================================
Define options
#===========================================================
set val(chan)          Channel/WirelessChannel ;      # channel type
set val(prop)          Propagation/TwoRayGround ;     # radio-propagation model
set val(ant)           Antenna/OmniAntenna ;          # Antenna type
set val(ll)            LL;                            # Link layer type
set val(ifq)           Queue/DropTail/PriQueue ;      # Interface queue type
set val(ifqlen)        250;                           # max packet in ifq
set val(netif)         Phy/WirelessPhy;               # network interface type
set val(mac)           Mac/802_11;                    # MAC type
set val(nn)            46                             # number of mobilenodes
set val(rp)            AODV                           # routing protocol
set val(x)             2000
set val(y)             2000
set opt(energymodel)   EnergyModel                    # Energy model
set opt(radiomodel)    RadioModel                     # Radio model
set opt(initialenergy) 100                            #Initial energy in Joules


Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set bandwidth_ 2e6
Phy/WirelessPhy set Pt_ 0.2818   ;# for 250.0

Phy/WirelessPhy set freq_ 914e+6
Phy/WirelessPhy set L_ 1.0

Mac/802_11 set PLCPDataRate_ 11.0e6       ;# 11Mbps
Agent/UDP set rate_ 8.0e6     ;                  # date rate 8Mbs

set ns [new Simulator]
#ns-random 0
set f [open 1_out.tr w]
$ns trace-all $f
set namtrace [open mobile5.nam w]
$ns namtrace-all-wireless $namtrace $val(x) $val(y)
$ns use-newtrace
```

```
set f0 [open n9-canpkts.tr w]

set f1 [open n9-canthr.tr w]

set f2 [open n9-canlat.tr w]

set f3 [open n9-energy.tr w]

set topo [new Topography]
$topo load_flatgrid 2000 2000

create-god $val(nn)
```

 # CONFIGURE AND CREATE NODES

```
$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) \
            -channelType $val(chan) \
            -topoInstance $topo \
            -agentTrace ON \
            -routerTrace ON \
            -macTrace ON \
            -movementTrace ON \
            -idlePower 0.06 \
                -rxPower 2.0 \
                -txPower 2.0 \
          -sleepPower 0.001 \
          -transitionPower 0.2 \
          -transitionTime 0.005 \

proc finish {} {

      global ns f f0 f1 f2 f3  namtrace
        $ns flush-trace
      close $namtrace
      close $f0
      close $f1
      close $f3
      exec xgraph Energy &
      exec xgraph Latency &
```

```
    exec xgraph NetworkLifetime &
    exec xgraph Throughput &

    exec nam -r 5m mobile5.nam &


 exit 0
}
set hrate1 0
set hrate2 0

proc record {} {
global sink0 sink1 sink2 sink3 sink4 sink5 sink6 sink7 sink8 sink9 sink10 sink11 sink12
sink13 sink14 sink15 sink16 sink17 sink18 sink19 sink20 f0 f1 f2 f2 f3  hrate1 hrate2

  #Get An Instance Of The Simulator
  set ns [Simulator instance]

  #Set The Time After Which The Procedure Should Be Called Again
  set time 0.05

  set can0 [$sink9 set npkts_]

  puts "No of Packets in CAN1 node : $can0"

  #set Size [expr 500 * 64 ]
  set Size 512

  set Thr  [expr $can0/$Size*10]

  set lt [expr (($can0+$hrate1)*8)/(2*$time*1000000)]

  puts " Latency in CAN1 node : $lt"

  set interval1 1.42

  set eng [expr $can0/$interval1]

  puts " Energy in the CAN1 node: $eng"


 #Get The Current Time
  set now [$ns now]

  #Save Data To The Files
```

```
 puts $f0 "$now [expr $can0]"
 puts $f1 "$now [expr $Thr]"
 puts $f2 "$now [expr $lt]"
 puts $f3 "$now [expr $eng]"

  #Re-Schedule The Procedure
 $ns at [expr $now+$time] "record"

 }
# define color index
$ns color 0 purple
$ns color 1 blue
$ns color 2 chocolate
$ns color 3 red
$ns color 4 brown
$ns color 5 tan
$ns color 6 gold
$ns color 7 black

set n(0) [$ns node]
$n(0) color "0"
$n(0) shape "circle"
set n(1) [$ns node]
$n(1) color "blue"
$n(1) shape "circle"
set n(2) [$ns node]
$n(2) color "tan"
$n(2) shape "circle"
set n(3) [$ns node]
$n(3) color "red"
$n(3) shape "circle"
set n(4) [$ns node]
$n(4) color "tan"
$n(4) shape "circle"
set n(5) [$ns node]
$n(5) color "red"
$n(5) shape "circle"
set n(6) [$ns node]
$n(6) color "0"
$n(6) shape "circle"
set n(7) [$ns node]
$n(7) color "brown"
$n(7) shape "circle"
set n(8) [$ns node]
$n(8) color "black"
$n(8) shape "circle"
```

```
set n(9) [$ns node]
$n(9) color "red"
$n(9) shape "circle"
set n(10) [$ns node]
$n(10) color "tan"
$n(10) shape "circle"
set n(11) [$ns node]
$n(11) color "red"
$n(11) shape "circle"
set n(12) [$ns node]
$n(12) color "red"
$n(12) shape "circle"
set n(13) [$ns node]
$n(13) color "tan"
$n(13) shape "circle"
set n(14) [$ns node]
$n(14) color "red"
$n(14) shape "circle"
set n(15) [$ns node]
$n(15) color "tan"
$n(15) shape "circle"
set n(16) [$ns node]
$n(16) color "red"
$n(16) shape "circle"
set n(17) [$ns node]
$n(17) color "red"
$n(17) shape "circle"
set n(18) [$ns node]
$n(18) color "tan"
$n(18) shape "circle"
set n(19) [$ns node]
$n(19) color "red"
$n(19) shape "circle"
set n(20) [$ns node]
$n(20) color "tan"
$n(20) shape "circle"
set n(21) [$ns node]
$n(21) color "red"
$n(21) shape "circle"
set n(22) [$ns node]
$n(22) color "red"
$n(22) shape "circle"
set n(23) [$ns node]
$n(23) color "tan"
$n(23) shape "circle"
set n(24) [$ns node]
```

```
$n(24) color "red"
$n(24) shape "circle"
set n(25) [$ns node]
$n(25) color "tan"
$n(25) shape "circle"
set n(26) [$ns node]
$n(26) color "red"
$n(26) shape "circle"
set n(27) [$ns node]
$n(27) color "red"
$n(27) shape "circle"
set n(28) [$ns node]
$n(28) color "tan"
$n(28) shape "circle"
set n(29) [$ns node]
$n(29) color "red"
$n(29) shape "circle"
set n(30) [$ns node]
$n(30) color "tan"
$n(30) shape "circle"
set n(31) [$ns node]
$n(31) color "red"
$n(31) shape "circle"
set n(32) [$ns node]
$n(32) color "red"
$n(32) shape "circle"
set n(33) [$ns node]
$n(33) color "tan"
$n(33) shape "circle"
set n(34) [$ns node]
$n(34) color "tan"
$n(34) shape "circle"
set n(35) [$ns node]
$n(35) color "tan"
$n(35) shape "circle"
set n(36) [$ns node]
$n(36) color "red"
$n(36) shape "circle"
set n(37) [$ns node]
$n(37) color "red"
$n(37) shape "circle"
set n(38) [$ns node]
$n(38) color "tan"
$n(38) shape "circle"
set n(39) [$ns node]
$n(39) color "tan"
```

```
$n(39) shape "circle"
set n(40) [$ns node]
$n(40) color "tan"
$n(40) shape "circle"
set n(41) [$ns node]
$n(41) color "red"
$n(41) shape "circle"
set n(42) [$ns node]
$n(42) color "red"
$n(42) shape "circle"
set n(43) [$ns node]
$n(43) color "tan"
$n(43) shape "circle"
set n(44) [$ns node]
$n(44) color "red"
$n(44) shape "circle"
set n(45) [$ns node]
$n(45) color "red"
$n(45) shape "circle"

for {set i 0} {$i < $val(nn)} {incr i} {
        $ns initial_node_pos $n($i) 30+i*100
}
```

 # subclass Agent/MessagePassing to make it do flooding

```
Class Agent/MessagePassing/Flooding -superclass Agent/MessagePassing

Agent/MessagePassing/Flooding instproc recv {source sport size data} {
   $self instvar messages_seen node_
   global ns BROADCAST_ADDR

   # extract message ID from message
   set message_id [lindex [split $data ":"] 0]
   #puts "\nNode [$node_ node-addr] got message $message_id\n"

   if {[lsearch $messages_seen $message_id] == -1} {
        lappend messages_seen $message_id
      $ns trace-annotate "[$node_ node-addr] received HELLO {$data} from $source"
      $ns trace-annotate "[$node_ node-addr] sending HELLO message $message_id"
        $self sendto $size $data $BROADCAST_ADDR $sport
   } else {

   }
}
Agent/MessagePassing/Flooding instproc send_message {size message_id data port} {
```

```
  $self instvar messages_seen node_
  global ns MESSAGE_PORT BROADCAST_ADDR

  lappend messages_seen $message_id
  $ns trace-annotate "[$node_ node-addr] sending HELLO message $message_id"
  $self sendto $size "$message_id:$data" $BROADCAST_ADDR $port
}
$ns at 0.0 "$n(0) setdest 710.682 516.271 3000.0"
$ns at 0.0 "$n(1) setdest 523.092 686.52 3000.0"
$ns at 0.0 "$n(2) setdest 98.1603 772.148 3000.0"
$ns at 0.0 "$n(3) setdest 625.528 866.461 3000.0"
$ns at 0.0 "$n(4) setdest 411.464 412.74 3000.0"
$ns at 0.0 "$n(5) setdest 510.118 505.858 3000.0"
$ns at 0.0 "$n(6) setdest 719.749 221.711 3000.0"
$ns at 0.0 "$n(7) setdest 745.043 863.836 3000.0"
$ns at 0.0 "$n(8) setdest 417.704 234.521 3000.0"
$ns at 0.0 "$n(9) setdest 766.905 372.481 3000.0"
$ns at 0.0 "$n(10) setdest 470.696 619.221 3000.0"
$ns at 0.0 "$n(11) setdest 66.0016 675.236 3000.0"
$ns at 0.0 "$n(12) setdest 158.29 392.946 3000.0"
$ns at 0.0 "$n(13) setdest 215.845 848.203 3000.0"
$ns at 0.0 "$n(14) setdest 187.888 770.282 3000.0"
$ns at 0.0 "$n(15) setdest 422.715 503.095 3000.0"
$ns at 0.0 "$n(16) setdest 659.152 675.423 3000.0"
$ns at 0.0 "$n(17) setdest 350.862 852.105 3000.0"
$ns at 0.0 "$n(18) setdest 580.377 203.216 3000.0"
$ns at 0.0 "$n(19) setdest 298.186 336.677 3000.0"
$ns at 0.0 "$n(20) setdest 668.314 416.218 3000.0"
$ns at 0.0 "$n(21) setdest 909.813 678.145 3000.0"
$ns at 0.0 "$n(22) setdest 591.354 385.571 3000.0"
$ns at 0.0 "$n(23) setdest 857.447 814.683 3000.0"
$ns at 0.0 "$n(24) setdest 619.701 586.512 3000.0"
$ns at 0.0 "$n(25) setdest 479.253 331.807 3000.0"
$ns at 0.0 "$n(26) setdest 657.931 251.018 3000.0"
$ns at 0.0 "$n(27) setdest 770.334 273.313 3000.0"
$ns at 0.0 "$n(28) setdest 351.304 642.512 3000.0"
$ns at 0.0 "$n(29) setdest 80.526 546.115 3000.0"
$ns at 0.0 "$n(30) setdest 140.606 642.651 3000.0"
$ns at 0.0 "$n(31) setdest 256.326 709.293 3000.0"
$ns at 0.0 "$n(32) setdest 348.918 737.454 3000.0"
$ns at 0.0 "$n(33) setdest 450.648 771.255 3000.0"
$ns at 0.0 "$n(34) setdest 253.048 543.828 3000.0"
$ns at 0.0 "$n(35) setdest 350.165 537.319 3000.0"
$ns at 0.0 "$n(36) setdest 261.912 649.548 3000.0"
$ns at 0.0 "$n(37) setdest 544.061 616.092 3000.0"
$ns at 0.0 "$n(38) setdest 400.263 580.653 3000.0"
```

```
$ns at 0.0 "$n(39) setdest 386.413 767.85 3000.0"
$ns at 0.0 "$n(40) setdest 491.557 737.678 3000.0"
$ns at 0.0 "$n(41) setdest 478.675 414.525 3000.0"
$ns at 0.0 "$n(42) setdest 166.839 478.046 3000.0"
$ns at 0.0 "$n(43) setdest 286.757 445.424 3000.0"
$ns at 0.0 "$n(44) setdest 174.853 578.926 3000.0"
$ns at 0.0 "$n(45) setdest 424.106 697.416 3000.0"

$ns at 30.0 "$n(9) setdest 622.107 501.814 50.0"
$ns at 43.0 "$n(9) setdest 710.682 516.271 50.0"
$ns at 43.0 "$n(0) setdest 604.602 479.902 50.0"

$ns at 53.0 "$n(9) setdest 468.318 668.703 150.0"
$ns at 59.0 "$n(9) setdest 523.092 686.52 5.0"
$ns at 59.0 "$n(1) setdest 406.335 646.291 8.0"

$ns at 0.1 "$n(17) color Brown"
$ns at 0.1 "$n(11) color Brown"

$ns at 0.1 "$n(7) color Red"
$ns at 0.1 "$n(23) color Red"

$ns at 0.1 "$n(3) color Red"
$ns at 0.1 "$n(21) color Red"


 $ns duplex-link $n(16) $n(7) 100Mb 30ms DropTail
 $ns duplex-link $n(16) $n(21) 100Mb 10ms DropTail
 $ns duplex-link $n(16) $n(3) 100Mb 2ms DropTail
 $ns duplex-link $n(16) $n(23) 100Mb 2ms DropTail

$ns at 0.1 "$n(16) color Blue"
$ns at 0.1 "$n(16) label Router"

$ns at 1.0 "$n(7) label BS"
$ns at 1.0 "$n(7) add-mark c1 green4 hexagon"
$ns at 1.0 "$n(21) add-mark c2 green4 hexagon"
$ns at 1.0 "$n(21) label BS"

$ns at 1.0 "$n(17) label EVENT"
$ns at 1.0 "$n(3) label BS"
$ns at 1.0 "$n(17) add-mark c4 green4 hexagon"
$ns at 1.0 "$n(3) add-mark c4 green4 hexagon"

$ns at 1.0 "$n(11) label EVENT"
$ns at 1.0 "$n(23) label BS"
```

```
$ns at 1.0 "$n(11) add-mark c5 green4 hexagon"
$ns at 1.0 "$n(23) add-mark c5 green4 hexagon"

$ns at 1.0 "$n(9) label CAN"
$ns at 0.1 "$n(9) color gold"
$ns at 1.0 "$n(9) add-mark c5 green4 hexagon"

$ns at 42.3 "$n(0) label Low-power"
$ns at 42.3 "$n(0) color red"

$ns at 55.8 "$n(1) label Low-power"
$ns at 55.8 "$n(1) color red"

$ns at 0.7 "$ns trace-annotate \"For 30 seconds mobiles nodes send HELLO messages to
establish 1-hop and 2-hop neighbors\""

# attach a new Agent/MessagePassing/Flooding to each node on port
$MESSAGE_PORT
for {set i 0} {$i < $num_nodes} {incr i} {
   set a($i) [new Agent/MessagePassing/Flooding]
   $n($i) attach  $a($i) $MESSAGE_PORT
   $a($i) set messages_seen {}
}


# now set up some events
$ns at 2.2 "$a(1) send_message 200 1 {first message}  $MESSAGE_PORT"
$ns at 2.8 "$a([expr $num_nodes/2]) send_message 600 2 {some big message}
$MESSAGE_PORT"
$ns at 3.4 "$a([expr $num_nodes-2]) send_message 200 3 {another one}
$MESSAGE_PORT"

# CONFIGURE AND SEexec nam -r 5m car.nam &T UP A FLOW

set sink0 [new Agent/LossMonitor]
set sink1 [new Agent/LossMonitor]
set sink2 [new Agent/LossMonitor]
set sink3 [new Agent/LossMonitor]
set sink4 [new Agent/LossMonitor]
set sink5 [new Agent/LossMonitor]
set sink6 [new Agent/LossMonitor]
set sink7 [new Agent/LossMonitor]
set sink8 [new Agent/LossMonitor]
set sink9 [new Agent/LossMonitor]
set sink10 [new Agent/LossMonitor]
set sink11 [new Agent/LossMonitor]
```

```
set sink12 [new Agent/LossMonitor]
set sink13 [new Agent/LossMonitor]
set sink14 [new Agent/LossMonitor]
set sink15 [new Agent/LossMonitor]
set sink16 [new Agent/LossMonitor]
set sink17 [new Agent/LossMonitor]
set sink18 [new Agent/LossMonitor]
set sink19 [new Agent/LossMonitor]
set sink20 [new Agent/LossMonitor]
set sink21 [new Agent/LossMonitor]
set sink22 [new Agent/LossMonitor]
set sink23 [new Agent/LossMonitor]
set sink24 [new Agent/LossMonitor]
set sink25 [new Agent/LossMonitor]
set sink26 [new Agent/LossMonitor]
set sink27 [new Agent/LossMonitor]
set sink28 [new Agent/LossMonitor]
set sink29 [new Agent/LossMonitor]
set sink30 [new Agent/LossMonitor]
set sink31 [new Agent/LossMonitor]
set sink32 [new Agent/LossMonitor]
set sink33 [new Agent/LossMonitor]
set sink34 [new Agent/LossMonitor]

$ns attach-agent $n(0) $sink0
$ns attach-agent $n(1) $sink1
$ns attach-agent $n(2) $sink2
$ns attach-agent $n(3) $sink3
$ns attach-agent $n(4) $sink4
$ns attach-agent $n(5) $sink5
$ns attach-agent $n(6) $sink6
$ns attach-agent $n(7) $sink7
$ns attach-agent $n(8) $sink8
$ns attach-agent $n(9) $sink9
$ns attach-agent $n(10) $sink10
$ns attach-agent $n(11) $sink11
$ns attach-agent $n(12) $sink12
$ns attach-agent $n(13) $sink13
$ns attach-agent $n(14) $sink14
$ns attach-agent $n(15) $sink15
$ns attach-agent $n(16) $sink16
$ns attach-agent $n(17) $sink17
$ns attach-agent $n(18) $sink18
$ns attach-agent $n(19) $sink19
$ns attach-agent $n(20) $sink20
$ns attach-agent $n(21) $sink21
```

```
$ns attach-agent $n(22) $sink22
$ns attach-agent $n(23) $sink23
$ns attach-agent $n(24) $sink24
$ns attach-agent $n(25) $sink25
$ns attach-agent $n(26) $sink26
$ns attach-agent $n(27) $sink27
$ns attach-agent $n(28) $sink28
$ns attach-agent $n(29) $sink29
$ns attach-agent $n(30) $sink30
$ns attach-agent $n(31) $sink31
$ns attach-agent $n(32) $sink32
$ns attach-agent $n(33) $sink33
$ns attach-agent $n(34) $sink34

set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0
set udp1 [new Agent/UDP]
$ns attach-agent $n(1) $udp1
set udp2 [new Agent/UDP]
$ns attach-agent $n(2) $udp2
set udp3 [new Agent/UDP]
$ns attach-agent $n(3) $udp3
set udp4 [new Agent/UDP]
$ns attach-agent $n(4) $udp4
set udp5 [new Agent/UDP]
$ns attach-agent $n(5) $udp5
set udp6 [new Agent/UDP]
$ns attach-agent $n(6) $udp6
set udp7 [new Agent/UDP]
$ns attach-agent $n(7) $udp7
set udp8 [new Agent/UDP]
$ns attach-agent $n(8) $udp8
set udp9 [new Agent/UDP]
$ns attach-agent $n(9) $udp9
set udp10 [new Agent/UDP]
$ns attach-agent $n(10) $udp10
set udp11 [new Agent/UDP]
$ns attach-agent $n(11) $udp11
set udp12 [new Agent/UDP]
$ns attach-agent $n(12) $udp12
set udp13 [new Agent/UDP]
$ns attach-agent $n(13) $udp13
set udp14 [new Agent/UDP]
$ns attach-agent $n(14) $udp14
set udp15 [new Agent/UDP]
$ns attach-agent $n(15) $udp15
```

```
set udp16 [new Agent/UDP]
$ns attach-agent $n(16) $udp16
set udp17 [new Agent/UDP]
$ns attach-agent $n(17) $udp17
set udp18 [new Agent/UDP]
$ns attach-agent $n(18) $udp18
set udp19 [new Agent/UDP]
$ns attach-agent $n(19) $udp19
set udp20 [new Agent/UDP]
$ns attach-agent $n(20) $udp20
set udp21 [new Agent/UDP]
$ns attach-agent $n(21) $udp21
set udp22 [new Agent/UDP]
$ns attach-agent $n(22) $udp22
set udp23 [new Agent/UDP]
$ns attach-agent $n(23) $udp23
set udp24 [new Agent/UDP]
$ns attach-agent $n(24) $udp24
set udp25 [new Agent/UDP]
$ns attach-agent $n(25) $udp25
set udp26 [new Agent/UDP]
$ns attach-agent $n(26) $udp26
set udp27 [new Agent/UDP]
$ns attach-agent $n(27) $udp27
set udp28 [new Agent/UDP]
$ns attach-agent $n(28) $udp28
set udp29 [new Agent/UDP]
$ns attach-agent $n(29) $udp29
set udp30 [new Agent/UDP]
$ns attach-agent $n(30) $udp30
set udp31 [new Agent/UDP]
$ns attach-agent $n(31) $udp31
set udp32 [new Agent/UDP]
$ns attach-agent $n(32) $udp32
set udp33 [new Agent/UDP]
$ns attach-agent $n(33) $udp33
set udp34 [new Agent/UDP]
$ns attach-agent $n(34) $udp34


proc attach-CBR-traffic { node sink size interval } {
  #Get an instance of the simulator
  set ns [Simulator instance]
  #Create a CBR sink14 agent and attach it to the node
  set cbr [new Agent/CBR]
  $ns attach-agent $node $cbr
```

```
  $cbr set packetSize_ $size
  $cbr set interval_ $interval

  #Attach CBR source to sink;
  $ns connect $cbr $sink
  return $cbr
 }
set cbr0 [attach-CBR-traffic $n(11) $sink31 512 .041]
set cbr1 [attach-CBR-traffic $n(31) $sink28 512 .041]
set cbr2 [attach-CBR-traffic $n(28) $sink15 512 .041]
set cbr3 [attach-CBR-traffic $n(15) $sink22 512 .041]
set cbr4 [attach-CBR-traffic $n(22) $sink9 512 .041]
set cbr5 [attach-CBR-traffic $n(9) $sink1 512 .041]
set cbr6 [attach-CBR-traffic $n(9) $sink16 512 .041]

set cbr7 [attach-CBR-traffic $n(17) $sink32 512 .041]
set cbr8 [attach-CBR-traffic $n(32) $sink1 512 .041]
set cbr9 [attach-CBR-traffic $n(1) $sink24 512 .041]
set cbr10 [attach-CBR-traffic $n(24) $sink20 512 .041]
set cbr11 [attach-CBR-traffic $n(20) $sink9 512 .041]
set cbr12 [attach-CBR-traffic $n(24) $sink9 512 .041]
set cbr13 [attach-CBR-traffic $n(9) $sink16 512 .041]

#set cbr14 [attach-CBR-traffic $n(16) $sink21 512 .041]
set cbr15 [attach-CBR-traffic $n(16) $sink23 512 .041]
set cbr16 [attach-CBR-traffic $n(16) $sink7 512 .041]
set cbr17 [attach-CBR-traffic $n(16) $sink3 512 .041]

set cbr18 [attach-CBR-traffic $n(9) $sink0 512 .041]
#set cbr19 [attach-CBR-traffic $n(18) $sink5 512 .041]
#set cbr20 [attach-CBR-traffic $n(18) $sink9 512 .041]

$ns at 0.0 "record"
$ns at 1.2 "$cbr0 start"
$ns at 1.2 "$cbr1 start"
$ns at 1.2 "$cbr2 start"
$ns at 1.2 "$cbr3 start"
$ns at 1.2 "$cbr4 start"
$ns at 55.2 "$cbr5 start"
$ns at 43.3 "$cbr6 start"

$ns at 15.0 "$cbr0 stop"
$ns at 15.0 "$cbr1 stop"
$ns at 15.0 "$cbr2 stop"
$ns at 15.0 "$cbr3 stop"
$ns at 15.0 "$cbr4 stop"
```

```
$ns at 69.0 "$cbr5 stop"
$ns at 53.0 "$cbr6 stop"

$ns at 15.2 "$cbr7 start"
$ns at 15.2 "$cbr8 start"
$ns at 15.2 "$cbr9 start"
$ns at 15.2 "$cbr10 start"
$ns at 15.2 "$cbr11 start"
$ns at 32.1 "$cbr12 start"
$ns at 59.2 "$cbr13 start"

$ns at 36.0 "$cbr7 stop"
$ns at 36.0 "$cbr8 stop"
$ns at 36.0 "$cbr9 stop"
$ns at 32.0 "$cbr10 stop"
$ns at 32.0 "$cbr11 stop"
$ns at 36.0 "$cbr12 stop"
$ns at 88.0 "$cbr13 stop"

#$ns at 47.0 "$cbr14 start"
$ns at 49.2 "$cbr15 start"
$ns at 51.3 "$cbr16 start"
$ns at 52.7 "$cbr17 start"

#$ns at 49.0 "$cbr14 stop"
$ns at 88.0 "$cbr15 stop"
$ns at 88.5 "$cbr16 stop"
$ns at 88.8 "$cbr17 stop"

$ns at 37.0 "$cbr18 start"
#$ns at 37.0 "$cbr19 start"
#$ns at 37.6 "$cbr20 start"

$ns at 43.0 "$cbr18 stop"
#$ns at 49.0 "$cbr19 stop"
#$ns at 54.0 "$cbr20 stop"

$ns at 95.0 "finish"
puts "Start of simulation.."
$ns run
```

REFERENCES

[1] Akyildiz I.F., Sankarasubramaniam W. Su, Y. and Cayirci E. (2002), "A Survey on Sensor Networks", IEEE Communications Magazine.

[2].K. Akayya and M. Younis, "An Energy-Aware QoS Routing Protocol for Wireless Sensor Networks", in the *Proceedings of IEEE Workshop* on *Mobile and Wireless Networks (MWN2003)*, Providence, RI, May 2003.

[3].D. Chen and P. K. Varshney, "QoS Support in Wireless Sensor Networks: A Survey", in the Proc. of the International Conference on Wireless Networks 2004 (ICWN'04), Las Vegas, Nevada, USA, June 21-24, 2004.

[4]. Juejia Zhou, Chundi Mu," A Kind of Application- Specific QoS Control in Wireless Sensor Networks", in the Proceedings of the International Conference on Information Acquisition 2006, Weihai, Shandong, China, August 20 - 23, 2006.

[5]. Liudong Xing and Akhilesh Shrestha", QoS Reliability of Hierarchical clustered Wireless Sensor Networks", in the proceedings of IEEE, 2006, pp 641-646.

[6]. C. S. Raghavendra, Krishna M. Sivalingam, Taieb F. Znati," Wireless sensor networks", Published by Springer ISBN 1402078838, 9781402078835.

[7]. Weike Chen, Wenfeng Li , Heng Shou, Bing Yuan, "A QoS-based Adaptive Clustering Algorithm for Wireless Sensor Networks", in the Proceedings of International Conference on Mechatronics and Automation, June 25 - 28, 2006, pp 1947-1952.

[8]. M. Younis, K. Akkaya, M. Eltoweissy, and A. Wadaa, "On Handling QoS Traffic in Wireless Sensor Networks", in the Proc. of 37th Hawaii International Conference on System Sciences (HICSS'04), pages: 292-301, Hawaii, Jan. 2004.

[9]. Yuanli Wang, Xianghui Liu, Jianping Yin, "Requirements of Quality of Service in Wireless Sensor Network", in the Proceedings of the International Conference on Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies (ICNICONSMCL'06), © 2006 IEEE.