Chapter 1

INTRODUCTION

1.1 Introduction to Wireless Sensor Network

Wireless sensor network (WSN) refers to a group of spatially dispersed and dedicated sensors for monitoring and recording the physical conditions of the environment and organizing the collected data at a central location. WSNs measure environmental conditions like temperature, sound, pollution levels, humidity, wind, and so on.

These are similar to wireless ad hoc networks in the sense that they rely on wireless connectivity and spontaneous formation of networks so that sensor data can be transported wirelessly. Sometimes they are called dust networks, referring to minute sensors as small as dust. Smart dust is a U C Berkeley project sponsored by DARPA. Dust Networks Inc., is one of the early companies that produced wireless sensor network products. WSNs are spatially distributed autonomous sensors to *monitor* physical or environmental conditions, such as temperature, sound, pressure, etc. and to cooperatively pass their data through the network to a main locations. The more modern networks are bi-directional, also enabling *control* of sensor activity. The development of wireless sensor networks was motivated by military applications such as battlefield surveillance; today such networks are used in many industrial and consumer applications, such as industrial process monitoring and control, machine health monitoring, and so on.

The WSN is built of "nodes" – from a few to several hundreds or even thousands, where each node is connected to one (or sometimes several) sensors. Each such sensor network node has typically several parts: a radio transceiver with an internal antenna or connection to an external antenna, a microcontroller, an electronic circuit for interfacing with the sensors and an energy source, usually a battery or an embedded form of energy harvesting. A sensor node might vary in size from that of a shoebox down to the size of a grain of dust, although functioning "motes" of genuine microscopic dimensions have yet to be created. The cost of sensor nodes is similarly variable, ranging from a few to hundreds of dollars, depending on the complexity of the individual sensor nodes. Size and cost constraints on sensor nodes result in corresponding constraints on resources such as energy, memory, computational speed and communications bandwidth. The topology of the WSNs can vary from a simple star network to an advanced multi-hop wireless mesh network. The propagation technique between the hops of the network can be routing or flooding.

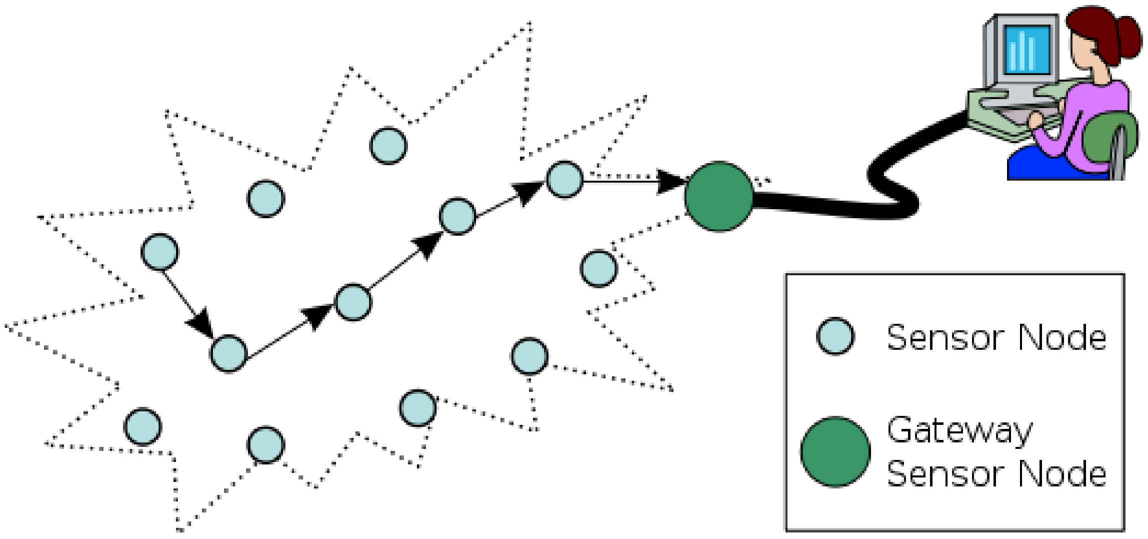


Fig 1.1:Wireless Sensor Network Architecture

Wireless Sensor Network (WSN) is used in different applications ranging from military to civilian applications including monitoring environmental conditions. WSN consists of a collection of large number of small devices randomly deployed with non-replinishable energy resources. Because of this limitation, routing in wireless sensor networks is a great challenge. Routing is challenging in wireless sensor networks since it cannot provide high message delivery ratio with little energy consumption . Routing protocols must ensure uniform energy consumption and provide secure routing among the sensor nodes thereby extending the sensor network lifetime . In spite of the aforementioned issues, since wireless sensor networks rely on wireless communication it can be easily attacked by the adversaries due to the open physical boundary of the network. Since the adversaries are well equipped they can perform malicious activities over the network such as jamming and trace-back attacks .

Wireless devices have a restricted transmission range for multi-hop communications, which becomes a real constraint when it exceeds the transmission range of nodes. WSNs represent an example of wireless networks that are emerging as latest trends among researches today because of their budding usages. A sensor network is defined as the constitution of a large number of nodes. The nodes of the network, referred to as sensor nodes, are battery-operated compact devices with the non-renewable energy resources. These sensors are normally deployed with uniform energy among them. But when it comes to the two divergent design issues for multi-hop wireless sensor networks, it relies on the energy balance and security.

Since all the nodes have non-rechargeable batteries so that the nodes die due to loss of energy, so lifetime optimization becomes a major issue while designing the network. Lifetime of the network mainly focuses on the message delivery ratio of the data packets among the sensor nodes. A large number of routing algorithms for WSNs have been emerged but most of them do not take into consideration the limited energy resources for sensor nodes. This is a main pitfall in major routing algorithms, where the routes are not selected based on the energy availability of nodes. This will not protract the lifetime of the sensor nodes and thus the network. There are various Topology-control algorithms including large amount of non-geographic ad hoc routing protocols proposed that are either proactive (maintain routes continuously) or reactive (create routes on-demand) while reducing energy consumption and improving the security levels of the network.

1.2 RCS Routing Protocol

Driven by the fact that WSN routing is most often geography based, we propose geography based efficient Resource Constraint Secure routing protocol for WSNs without relying on the concept of flooding. In CASER the messages are transmitted using random walking routing strategy which directs the chance of choosing the nodes with low energy as relay nodes at times. To keep away from this, the data is transmitted via energy aware route only and the MES scheme on Elliptic curve algorithm is used to afford authentication. Elliptic curve cryptography is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields.

One of the main benefits in comparison with non-ECC cryptography (with plain Galois fields as a basis) is the same level of security provided by keys of smaller size. For the sake of security the message content is encrypted using a secret key encryption technique and decrypted at the sink node by knowing the same secret key used by the source. Since the unauthenticated person cannot access the original data, the protocol provides a secure message delivery opportunity to increase the message delivery ratio in the presence of attacks.

RCS routing protocol ensures balanced energy consumption and secure routing. Also, routing trace-back attacks and hostile traffic jamming attacks can be prevented. The main objective of RCS routing protocol is to provide energy aware and secure routing for message in a wireless sensor network with fixed nodes and to increase the lifetime of network. Our work uses the energy efficient routing and also provides the hop-by-hop security by using the MES Scheme. Our work does not make use of the random walking technique of routing which is being used in the CASER. Since, the random walking technique is not being used in RCS routing protocol, there is no chance of choosing the low energy node as relay node and hence the energy balance of the whole network can be maintained. The authentication is being done using the Modified ElGammal Signature scheme. Along with this signature the security for data packet is provided to detect the adversaries. The message receiver should be able to verify whether the message sent by the authorized node and also verify whether the message has been modified by the adversaries. Every forwarder can verify the message is authenticated or not. In this scheme, the authentication of the message is checked during each and every hop and therefore the security process is a bit complicated.

The mechanism of the proposed routing protocol is explained in Fig. 1. In RCS the network is divided into many grids where in each grid is governed by a Grid Head (*gh*). The GH is selected based on the residual energy of the nodes in the network. The node which is having the highest residual energy within the grid is chosen as the GH of the corresponding grid. In this routing protocol the source grid identifies the destination grid as follows. Source grid(*sg*) forwards the message sent by its member to one of its neighbouring grid(*ng*). The GH which is closer to the destination grid (*ddg*) and whose residual energy *(ghre*) is greater than the rest of the GHs is chosen as the neighbouring grid. This information is appended to the Shortest Route (*sr*). Message authentication between the Grid Heads is performed by Signature Generation and Verification. The process is continued until the destination grid (*dg*) is reached.

The GHs are periodically re-elected based on their residual energy level and the Threshold (*th*) value. Threshold is the constraint which is enforced in the grid head selection which is nothing but the average residual energy of all the nodes within the grid at a given point of time. This feature of GH selection leads to the maximization of the sensor network lifetime.

1.3 Motivation

Motivation behind this project is to minimize the power consumption of the sensor nodes and increase the life time of network thus ensuring the efficiency of the Wireless sensor network. Wireless sensor network is used among various application fields like military applications, forest application and so on. Wireless sensor network consist of large collection of devices known as sensor nodes associated with non-replinishable energy resources. In wireless sensor networks the rate of message delivery does depend on the energy consumption. The sensor nodes in wireless sensor network are battery operated compact devices with the non-renewable energy resources which are all randomly deployed with uniform energy among them. The routes of data transmission should have been selected based on the energy consumption. The Motivation was to design a protocol known as resource constraint secure protocol which may be an appropriate protocol in order to save the residual energy and also helps in the transmission of data from one node to another in a more efficient manner as it targets the nodes which has higher residual energy.

1.4 Problem Statement

To design and implement the system which saves the energy of various nodes and also helps to transfer data in shortest route possible. But the problem is, if any nodes goes out of power that is, if any nodes loses their residual energy it won't be able to function furthermore. The input for our proposed model will be adding nodes into the server with specified residual energy for each node.

1.5 Scope of the project

- An efficient protocol which makes appropriate transfer of data from one node to another.
- The algorithm that reduces the power consumption as well as residual energy of nodes by checking battery status of each node.

- The protocol is responsible for minimizing the power consumption.
- It is specially used in places where we need the node to work for longer period of time. for e.g. military fields and forest etc.,

1.6 Objectives

The main objective of the project is to design a routing protocol ,

- To maximize the Overall Lifetime of the Network.
- Which makes the efficient transfer of data with shortest route possible.
- To ensure that the data sent by source node and received at the destination node are authenticated.
- To minimize the residual energy consumption.

1.7 Review of Literature

Wenyuan Xu, Ke Ma, Wade Trappe and Yanyong Zhang proposed Jamming sensor networks: attack and defense strategies in 2006-IEEE Network. [4].

Wireless sensor networks are built upon a shared medium that makes it easy for adversaries to conduct radio interference, or jamming, attacks that effectively cause a denial of service of either transmission or reception functionalities. These attacks can easily be accomplished by an adversary by either bypassing MAC-layer protocols or emitting a radio signal targeted at jamming a particular channel. In this article we survey different jamming attacks that may be employed against a sensor network. In order to cope with the problem of jamming, we discuss a two-phase strategy involving the diagnosis of the attack, followed by a suitable defense strategy. We highlight the challenges associated with detecting jamming. To cope with jamming, we propose two different but complementary approaches. One approach is to simply retreat from the interferer, which may be accomplished by either spectral evasion (channel surfing) or spatial evasion (spatial retreats). The second approach aims to compete more actively with the interferer by adjusting resources, such as power levels and communication coding, to achieve communication in the presence of the jammer.

Securing sensor networks is a challenging task due to the limited resources associated with low-cost sensor hardware. The combination of the commodity nature of wireless technologies and an increasingly sophisticated user base means that adversaries are able to easily gain access to communications between sensor devices by purchasing their own device and running it in a monitor mode. Conventional cryptographic security mechanisms are being translated to the sensor domain in order to defend against attacks like packet injection and spoofing network level control information. However, in spite of the progress being made to apply network security in the sensor realm, sensor networks will remain vulnerable to attacks that target their use of the wireless medium.

Fraser Cadger, Kevin Curran, Jose Santos and Sandra Moffett proposed A survey of geographical routing in wireless ad-hoc networks. IEEE Communications Surveys & Tutorials in 2013.[5].

Geographic routing offers a radical departure from previous topology-dependent routing paradigms through its use of physical location in the routing process. Geographic routing protocols eliminate dependence on topology storage and the associated costs, which also makes them more suitable to handling dynamic behavior frequently found in wireless ad-hoc networks. Geographic routing protocols have been designed for a variety of applications ranging from mobility prediction and management through to anonymous routing and from energy efficiency to QoS. Geographic routing is also part of the larger area of context awareness due to its usage of location data to make routing decisions and thus represents an important step in the journey towards ubiquitous computing. The focus of this system, within the area of geographic routing is on wireless ad-hoc networks and how location information can benefit routing. This paper aims to provide both a comprehensive and methodical survey of existing literature in the area of geographic routing from its inception as well as acting as an introduction to the subject.

WIRELESS ad-hoc networks (which shall be referred to in this paper as ad-hoc networks) are a field of networking in which networks are formed when required and typically for short durations. Ad-hoc networks are typically decentralized and do not feature dedicated devices with defined roles such as routers or switches. Instead all participating nodes act as both routers and end-users. As devices are limited by their radio range ad-hoc networks typically employ a strategy known as multi-hopping in which a source node will send a message to the destination by passing it to a series of intermediate node. This enables geographically disparate nodes to communicate wirelessly. Multi-hopping is typical of the distributed architecture of ad-hoc networks, and one of its biggest advantages. As ad-hoc networks use multi-hopping and do not rely on infrastructure they can be deployed anywhere two or more devices that share a suitable communications medium (WiFi, Bluetooth, UWB, etc.) are present. The field of ad-hoc networking itself contains several subfields such as Mobile Ad-Hoc Networks (MANETs) where all nodes are assumed to be mobile, Wireless Mesh Networks (WMNs) a combination of ad-hoc and infrastructure network, Wireless Sensor Networks (WSNs) ad-hoc networks made up of small sensor devices, and Vehicular Ad-hoc Networks (VANETs).

Di Tang, Tongtong Li, Jian Ren and Jie Wu, "Cost-aware SEcure routing (CASER) protocol design for wireless sensor networks IEEE Transactions on Parallel and Distributed Systems",2015.[6].

Lifetime optimization and security are two conflicting design issues for multi-hop wireless sensor networks (WSNs) with non-replinishable energy resources. In this paper, we first propose a novel secure and efficient Cost-Aware SEcure Routing (CASER) protocol to address these two conflicting issues through two adjustable parameters: energy balance control (EBC) and probabilistic based random walking. We then discover that the energy consumption is severely disproportional to the uniform energy deployment for the given network topology, which greatly reduces the lifetime of the sensor networks. To solve this problem, we propose an efficient non-uniform energy deployment strategy to optimize the lifetime and message delivery ratio under the same energy resource and security requirement. We also provide a quantitative security analysis on the proposed routing protocol. Our theoretical analysis and OPNET simulation results demonstrate that the proposed CASER protocol can provide an excellent trade-off between routing efficiency and energy balance, and can significantly extend the lifetime of the sensor networks in all scenarios. For the non-uniform energy deployment, our analysis shows that we can increase the lifetime and the total number of messages that can be delivered by more than four times under the same assumption. We also demonstrate that the proposed CASER protocol can achieve a high message delivery ratio while preventing routing trace back attacks.

The recent technological advances make wireless sensor networks (WSNs) technically and economically feasible to be widely used in both military and civilian applications, such as monitoring of ambient conditions related to the environment, precious species and critical infrastructures. A key feature of such networks is that each network consists of a large number of untethered and unattended sensor nodes. These nodes often have very limited and non-replinishable energy resources, which makes energy an important design issue for these networks.

Zakhary Same, Radenkovic Milena and Benslimane Abderrahim (2014) Efficient location privacy-aware forwarding in opportunistic mobile networks. IEEE Transactions on Vehicular Technology.[8].

This system proposes a novel fully distributed and collaborative k-anonymity protocol (LPAF) to protect users' location information and ensure better privacy while forwarding queries/replies to/from un-trusted Location-based Service (LBS) over opportunistic mobile networks (OppMNet). We utilize a lightweight multi-hop Markov-based stochastic model for location prediction to guide queries towards the LBS's location as well as to reduce required resources in terms of retransmission overheads. We develop a formal analytical model and present theoretical analysis and simulation of the proposed protocol performance. Validate our results by performing extensive simulation experiments over pseudo realistic city-map using map-based mobility models and using real-world data trace to compare LPAF to existing location privacy and benchmark protocols and show that LPAF manages to keep higher privacy levels in terms of k-anonymity, and quality of service in terms of success ratio and delay, compared to other protocols while maintaining lower overheads. Simulation results show that LPAF achieves up to 11% improvement in success ratio for pseudo realistic scenarios, while real-world data trace experiments show up to 24% improvement with a slight increase in the average delay.

This system is concerned with the source k-anonymity location-privacy when contacting an LBS in OppMNet through obfuscation. Obfuscation refers to the collaborative activities by nodes to deliberately degrade the quality of information collected by the LBS about the source of the LBS query. We focus on a class of LBSs that does not require user identity in order to provide the service. Other anonymity and security aspects in OppMNet are outside the scope of this paper. We utilize a stochastic model for location predication, and propose a lightweight Markov model to drive the privacy preserving protocol. We recognize two possibilities based on whether the nodes are capable of knowing their exact location (such as GPS coordinates). For the first case, where the nodes are able to detect their exact location coordinates, inspired by [7], we propose a path prediction maintained locally at the individual nodes using Markov model proximity. As for the second case, where the nodes are unable to determine their exact location coordinates, we use the recorded Ids for the sighting of fixed infrastructure points (either access-points or GSM cell IDs) as the location identifier. Unlike [7], our proposed model utilizes multihop prediction. Nodes exchange their local predictions and that of their own friends when they meet each other, during opportunistic encounters, to help obtain more accurate/updated prediction estimate through shared knowledge exchanged in a distributed way.

Al-Sakib Khan Pathan, Hyung-Woo Lee and Choong Seon Hong, "Security in wireless sensor networks: issues and challenges", 2006.[9]

Wireless Sensor Network (WSN) is an emerging technology that shows great promise for various futuristic applications both for mass public and military. The sensing technology combined with processing power and wireless communication makes it lucrative for being exploited in abundance in future. The inclusion of wireless communication technology also incurs various types of security threats. The intent of this paper is to investigate the security related issues and challenges in wireless sensor networks. We identify the security threats, review proposed security mechanisms for wireless sensor networks. We also discuss the holistic view of security for ensuring layered and robust security in wireless sensor networks.

Haibo Zhang and Hong Shen," Balancing energy consumption to maximize network lifetime in data-gathering sensor networks". IEEE-2009.[10].

Unbalanced energy consumption is an inherent problem in wireless sensor networks characterized by multi hop routing and many-to-one traffic pattern, and this uneven energy dissipation can significantly reduce network lifetime. In this paper, we study the problem of maximizing network lifetime through balancing energy consumption for uniformly deployed data-gathering sensor networks. We formulate the energy consumption balancing problem as an optimal transmitting data distribution problem by combining the ideas of corona-based network division and mixed-routing strategy together with data aggregation. Propose a localized zone-based routing scheme that guarantees balanced energy consumption among nodes within each corona. We then design an offline centralized algorithm with time complexity O(n) (n is the number of coronas) to solve the transmitting data distribution problem aimed at balancing energy consumption among nodes in different coronas. The approach for computing the optimal number of coronas in terms of maximizing network lifetime is also presented. Based on the mathematical model, an energy-balanced data gathering (EBDG) protocol is designed and the solution for extending EBDG to large-scale data-gathering sensor networks is also presented. Simulation results demonstrate that EBDG significantly outperforms conventional multi hop transmission schemes, direct transmission schemes, and cluster-head rotation schemes in terms of network lifetime.

Feng Liu, Chi-Ying Tsui and Ying Jun Zhang," Joint routing and sleep scheduling for lifetime maximization of wireless sensor networks". IEEE-2009.[14].

The rapid proliferation of wireless sensor networks has stimulated enormous research efforts that aim to maximize the lifetime of battery-powered sensor nodes and, by extension, the overall network lifetime. Most work in this field can be divided into two equally important threads, namely (i) energy efficient routing that balances traffic load across the network according to energy-related metrics and (ii) sleep scheduling that reduces energy cost due to idle listening by providing periodic sleep cycles for sensor nodes. To date, these two threads are pursued separately in the literature, leading to designs that optimize one component assuming the other is pre-determined. Such designs give rise to practical difficulty in determining the appropriate routing and sleep scheduling schemes in the real deployment of sensor networks, as neither component can be optimized without pre-fixing the other one. This paper endeavors to address the lack of a joint routing-and-sleep-scheduling scheme in the literature by incorporating the design of the two components into one optimization framework. Notably, joint routing and sleep scheduling by itself is a non-convex optimization problem, which is difficult to solve. We tackle the problem by transforming it into an equivalent Signomial Program (SP) through relaxing the flow conservation constraints. The SP problem is then solved by an iterative

Geometric Programming (IGP) method, yielding an near optimal routing-and-sleep-scheduling scheme that maximizes network lifetime.

To the best of our knowledge, this is the first attempt to obtain the optimal joint routing and sleep-scheduling strategy for wireless sensor networks. The near optimal solution provided by this work opens up new possibilities for designing practical and heuristic schemes targeting the same problem, for now the performance of any new heuristics can be easily evaluated by using the proposed near optimal scheme as a benchmark.

1.7 Organization of the report

The project report is organized as follows

Chapter-1 Introduction

This chapter presents the brief introduction on Wireless sensor network, problem statement, objectives, scope of the project, and literature survey.

Chapter-2 System requirement specification

This chapter presents the various software and hardware requirements. It also presents a brief description of the software used.

Chapter-3 High level design

This chapter presents the system architecture of proposed system and data flow diagram of each module.

Chapter-4 Detailed design

This chapter presents the flowchart diagram and detailed functionality and processing of each module.

Chapter-5 Implementation

This chapter explains about the implementation requirements programming language selection and also coding lines for programming language used in the project.

Chapter-6 Testing

This chapter deals with the software test environment, test procedures and test cases for each module.

Chapter-7 Results and Discussion

This chapter presents snapshots with respect to 'novel resource constraint secure routing protocol for wireless sensor network.

1.8 Summary

This chapter gives introduction to the project, brief introduction about Wireless sensor network and also about the resource constraint secure routing protocol, the scope of the project, problem statement and objectives of the project, the related paper presented in the literature survey and also finally the organization of the report

Chapter 2

SYSTEM REQUIREMENTS SPECIFICATION

2.1 Specific Requirements

System requirements specification is a detailed statement of the effects that a system is require achieving. A good specification gives a complete statement of what the system is to do, without making any commitment as to how the system is to do it. It constraints only the externally observable behavior and omits any design or implementation.

2.2 Hardware Requirements:

System	:	Pentium IV 2.4 GHz or higher
Hard Disk	:	500 GB or higher
Ram	:	4 GB or higher

- Any desktop / Laptop system with above configuration or higher level

2.3 Software Requirements:

Operating system	:	Windows XP / 7 / 8 / higher
Programming Language	:	Java (Jdk 1.7)
IDE	:	Eclipse

2.4 Summary

In this chapter system requirement specification contains hardware requirements and software requirements.

Chapter 3

HIGH LEVEL DESIGN

A software product is a complex entity. Its development usually follows what is known as Software Development Life Cycle (SDLC).The second stage in the SDLC is the Design stage. The objective of the design stage is to produce the overall design of the software. In the High-Level Design, the proposed functional and non-functional requirements of the software are studied. Overall solution architecture of the solution is developed which can handle those needs.

3.1 Design Considerations

Design considerations for efficient transmission of data from one node to another using Resource Constraint Secure protocol for wireless sensor networks.

- predictive techniques are used such as transmitting the data via energy aware route only.
- Geographic routing protocols are the most preferred routing protocols for Wireless Sensor Networks since they rely on geographic position information.
- The ability to select the nodes based on high residual energy which in turn results in the faster mode of transmission.

3.2 System Architecture

System architecture is a conceptual model that defines the structure behavior and more views of a system. It can comprise system components that will work together to implement the overall system.

The Fig 3.1 represents the system architecture of Resource constraint secure protocol. Initially all the nodes are connected to the server. Then the network is divided into two or more equal sized grids . Each grid does contain a cluster head which is elected based on higher residual energy i.e., the node with higher residual energy is elected as the cluster head in each grid . In which many cluster nodes are interconnected to the cluster head and the data is transmitted through the nodes to the cluster head. The data sent to the cluster head of that grid sends the data to another cluster head of the other grid then transfers the same data to the base station, Otherwise data directly goes to the base station.

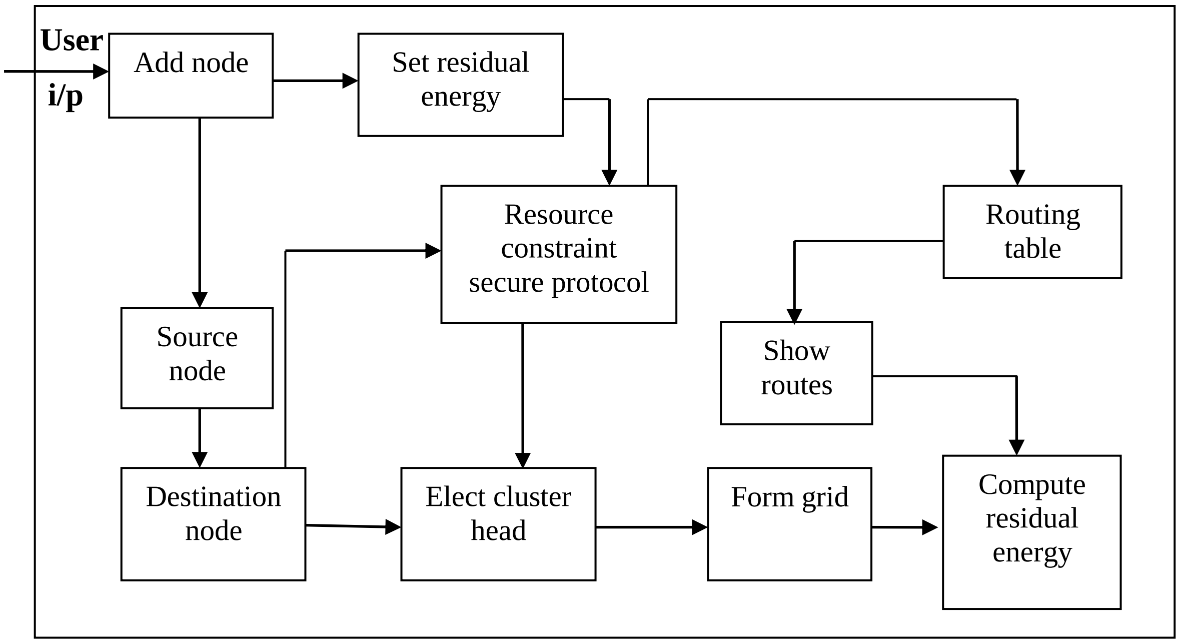


Fig 3.1: System Architecture Of RCS Protocol.

3.3 Specification Using Use Case Diagram

A use case diagram in the unified module language (UML) is a type of behavioral diagram defined by and created from a use case analysis. Its purpose is to present the graphical overview of the functionality provided by the system in terms of actors, their goals (represented as use cases) and any dependencies between those use cases. The main purpose of the use case diagram is to show what system functions performs for which actors. Roles of the actors in the system can be depicted.

Diagram Building Blocks:

- **Use cases-** A use case describes a sequence of actions that provide something of measurable value to an actor and is drawn in horizontal ellipse.
- **Actors-** An actor is a person, organization or an external system that plays a role in one or more interactions with the system.

Both consist of different modules to define the overall system of the process to design form of the diagram to show how to process the implementation of the running method of the project to be identified of the other to compare the process of the methods.

3.3.1 Use case diagram for Setting Residual Energy

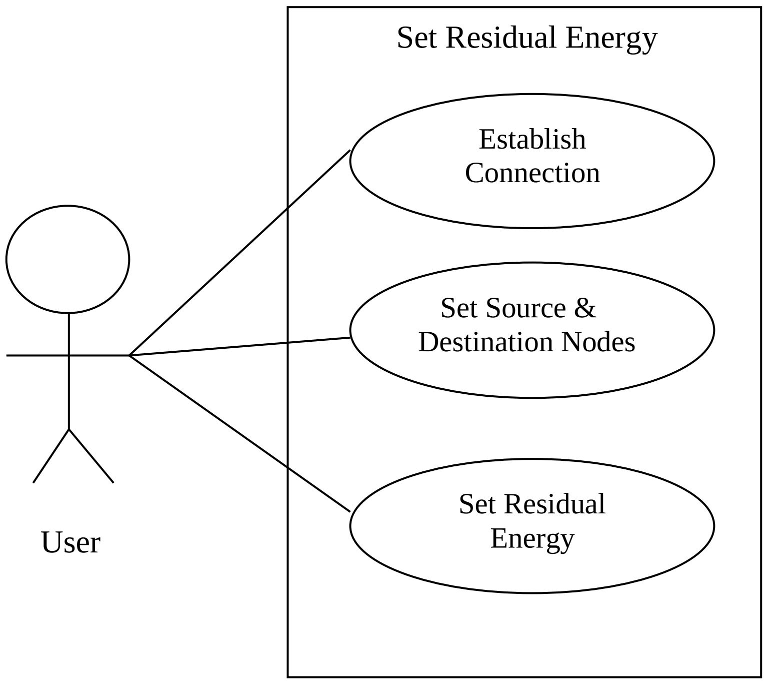


Fig 3.2:Use Case Diagram For Setting Residual Energy.

The Fig 3.2 represents the use case diagram of setting the node residual energy. Initially the nodes are created that is the nodes are added to the server, then the connection among those nodes are established in multi direction. Many number of nodes can be added into the network and connection among them would be established in parallel. Initially the residual energy of several nodes is set to 10. The residual energy of those nodes remain as 10 even after the 1st iteration and it gradually decreases after then. So after multiple iterations we can see the decrease in residual energy.

3.3.2 Use case diagram for Transfer of Data

The Fig 3.3 represents the use case diagram for the transfer of data . In this module once the nodes are added and as the connection is established. we have to select the source node and destination node for the transfer of data packets. Once we select the source and destination nodes we have to elect the cluster head. The election of cluster head is a random process that is the cluster head is elected based on the highest residual energy and after the election of cluster head formation of grid based on source and destination nodes then the data can be sent in shortest route possible and also minimizes the energy consumption of nodes.

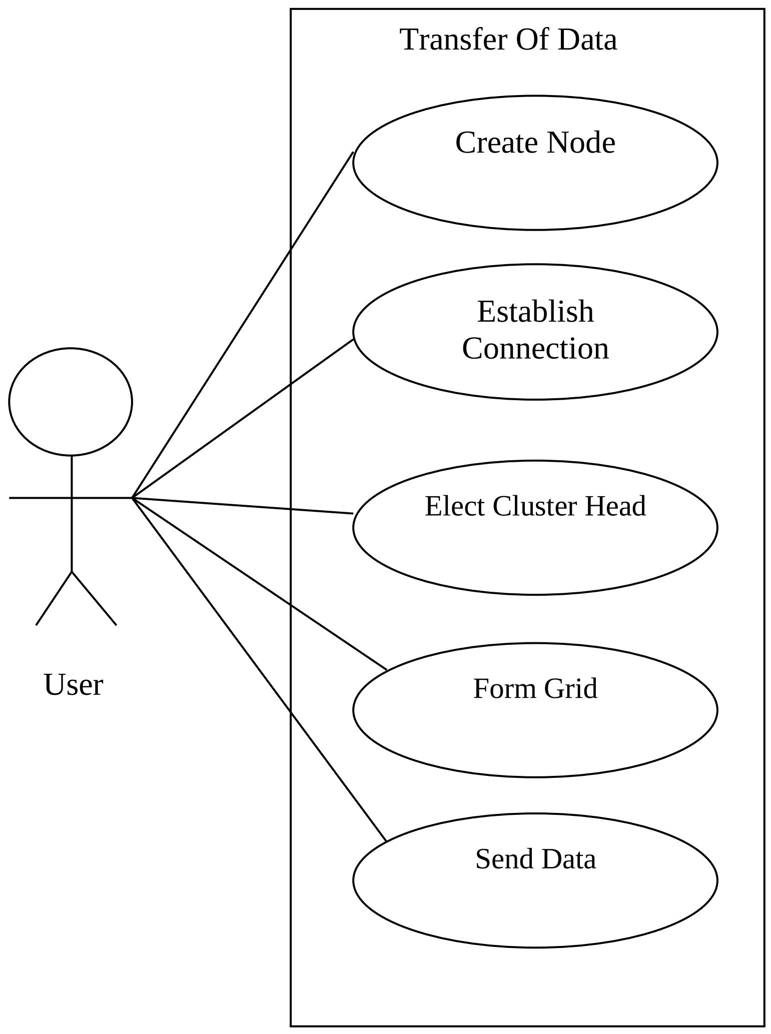


Fig 3.3: Use case diagram for Transfer of Data.

3.3.3 Use case diagram for Computation of Residual Energy

The Fig 3.4 represents the use case diagram for the computation of residual energy. In this module, as we add nodes into the server and establish connection then we do the transfer of data packets from one node to another through the election of cluster nodes and by forming equal grids. When the data packets are transferred for multiple number of times, the residual energy of the sensor nodes goes down. By making use of the resource constraint secure protocol the enhancement is seen in terms of efficiency that is reducing the consumption of residual energy of each node. Hence after multiple iterations we can see the change in residual energy of nodes and for this we have to compute the residual energy after multiple set of operations.

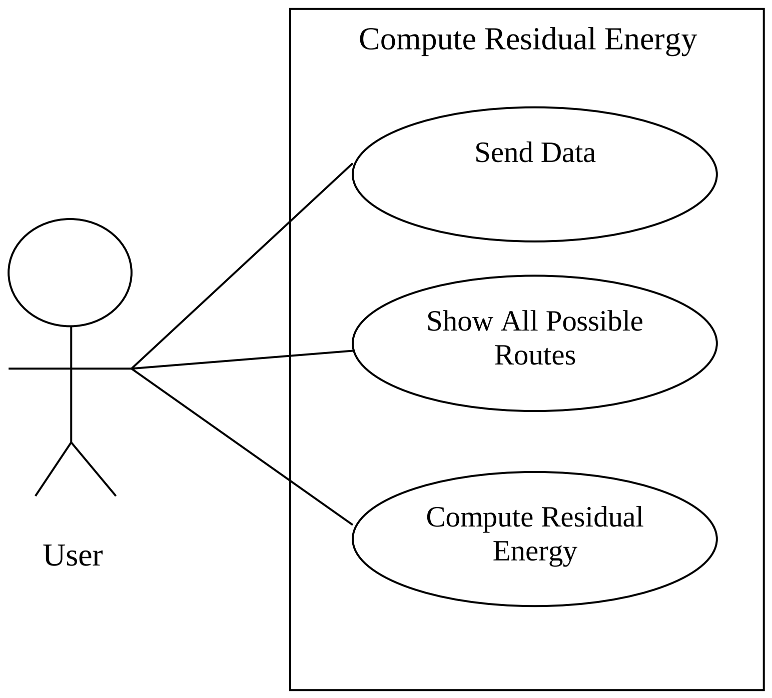


Fig 3.4: Use case diagram for Computation of Residual Energy.

3.3.4 Use case diagram for Generating Reports and Routing Table

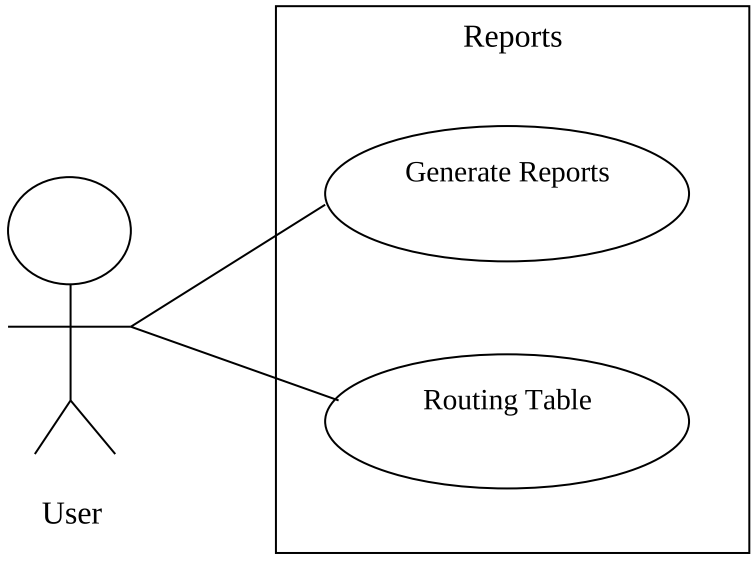


Fig 3.5: Use case diagram for Generating Reports and Routing Table.

The Fig 3.5 represents the Use case diagram for the Generation of Reports and Routing Table. As and when the data packets are getting transferred from one node to another node, the routing table is computed and also the report is generated. The Report consists of graphs which is used for the comparison between the existing system and the proposed system.

3.3.4 Use case diagram for Entire System

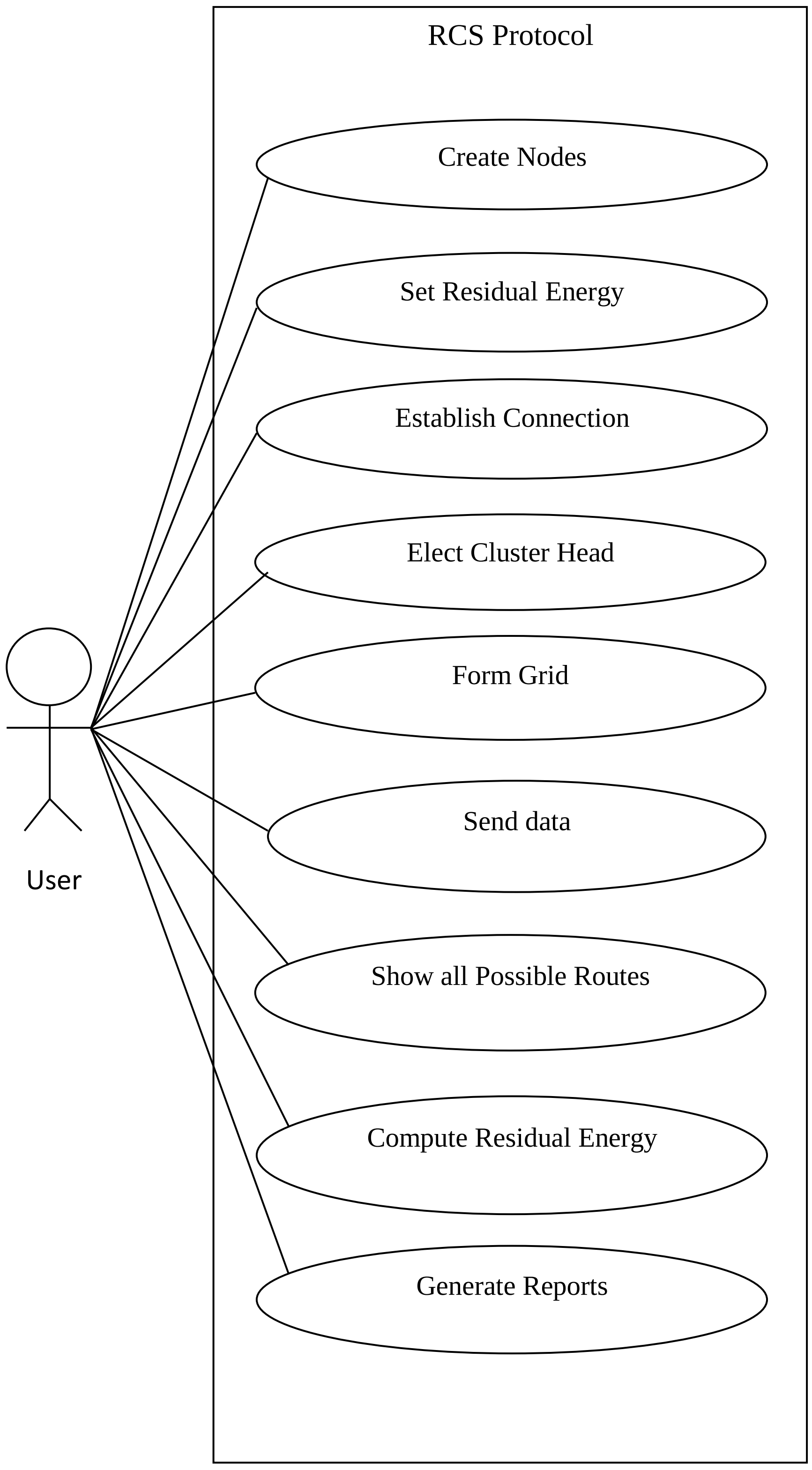


Fig 3.6: Use case diagram for Entire system.

The Fig 3.6 represents the use case diagram of the entire system. This contains the detailed description of what all happens in the proposed project. Once the nodes are been created and as the connection between them is established. The residual energy of every nodes is set to a particular value which remains same in the first iteration and goes down after a set of iterations. Then the source and destination nodes are selected for the transfer of data packets followed by the selection of cluster head, which indeed is a random process but surely does select nodes which has got higher residual energy. As the process of election of cluster head takes place, the grid formation takes place. Finally the data packets are being sent based on the election of cluster head that is the data packets from source node to the cluster head of that particular grid to the cluster head of another grid and from there it is passed on to the destination node. The data transfer from source to destination takes place as discussed above by the use of resource constraint secure protocol also makes sure that the consumption of residual energy is minimized. The reports are also generated which shows the consumption of residual energy of sensor nodes.

3.4 Data Flow Diagram

A Data Flow Diagram (DFD) is a graphical representation of flow of data through an information request, modelling its process aspects, A DFD is often used as preliminary step to create an overview of the system, which can later be elaborated.

DFDs can be used for visualization of data processing (structured design).A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel.

3.4.1 Physical DFD

A Physical DFD shows how the system is actually implemented, either at the moment (current physical DFD), or how the designer intends it to be in the future Required Physical DFD. Thus a physical DFD may be used to describe the set of data items that appear on each piece of paper.

The Fig 3.4 represents the physical data flow diagram of resource constraint secure routing protocol. As shown in the above figure the data packets are always sent from the source node and by RCS protocol the data is sent in shortest route possible i.e.

the data packets from the source node to the cluster head of one of the grid and then the same data packets are transferred to the cluster head of another grid. Here the data packets doesn’t go through all the intermediate nodes. Instead it assigns some nodes as cluster heads and the election of cluster head is based on the higher residual. Higher the residual energy, more is the chances for it to be the cluster head. Hence the data collected at the cluster head of the other grid is finally passed on to the destination grid.

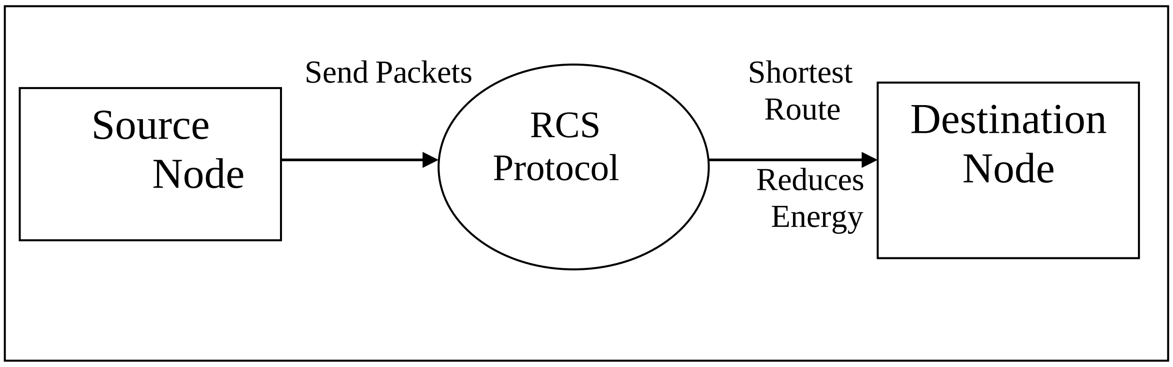


Fig 3.4: Physical Data flow diagram showing Resource constraint secure protocol.

The Fig 3.5 shows a data flow diagram of Resource constraint secure routing protocol. Upon the selection of add node option, nodes with sufficient residual energy will be created followed by the computation of the residual energy of each node. As and when the nodes are added into the server the connection among those nodes are established simultaneously. Then source node and destination node is selected to make the transfer of data. The data packets are being sent from source node to the cluster head of one of the grid from there, data is passed on to the cluster head of the other grid. The grids will be consisting of equal number of nodes. The nodes with the highest residual energy will be elected as the cluster heads. The rest of the nodes will be marked as the common nodes and will be connected to the cluster head of their respective grid. The information sent from the source node will be forwarded to its cluster head, then it is broadcasted to the cluster head of the destination node's grid and finally the information is delivered to the intended destination node. As and when the data is transferred from one node to another the route is traced and the neighboring node for that route is displayed in the routing table and also in the routing log all the information of transfer of data through various nodes and different time intervals is displayed. When the report is generated the graph is shown which gives the comparison between the existing system and proposed system which shows the amount of energy used and also by using RCS protocol more amount of energy is saved in the proposed system.

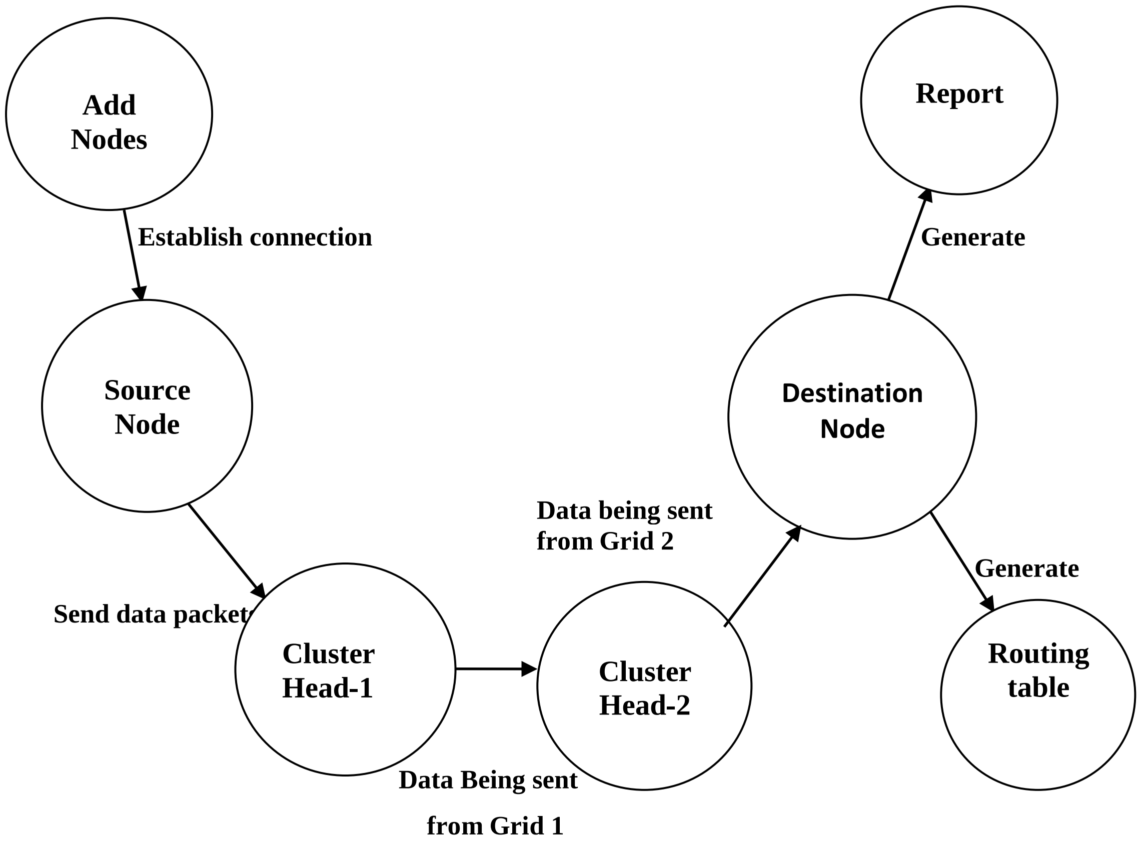


Fig 3.5: Data flow diagram of the proposed system.

3.5 State Chart Diagram

The state diagrams are used to give an abstract description of behavior of a system. This behavior is analyzed and represented in series of events that could occur in one or more possible states. The initial state in state diagram is represented in solid circle. State diagram is the diagram used in computer science to describe the behavior of a system considering all the possible states of an object when an event occurs. The fig 3.6 shows the state diagram of the entire system i.e. proposed system.Once the nodes are added into the network and as they are connected to the server, the nodes with higher residual energy is selected. The residual energy is computed and if the computed residual energy is more than the scheduled energy then that node is selected as the cluster head. The information or data which are been sent by the source grid is given to the cluster head and the cluster heads information is being broadcasted and finally it reaches the destination grid and if the computed residual energy of the node is lesser than the scheduled energy then we have to mark that node as a common node.

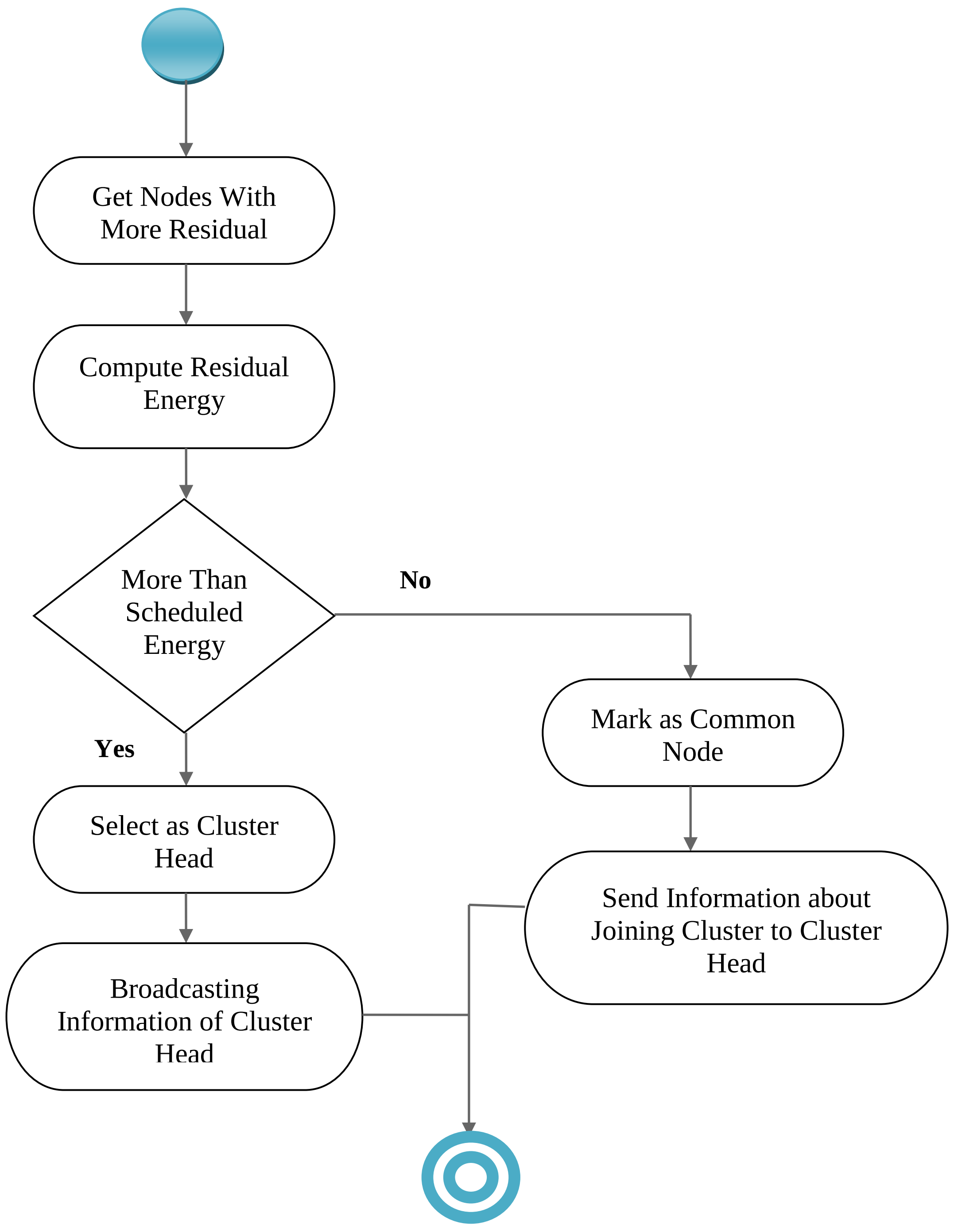

Get Nodes With More Residual
Compute Residual Energy
More Than Scheduled Energy
No
Yes
Mark as Common Node
Select as Cluster Head
Send Information about Joining Cluster to Cluster Head
Broadcasting Information of Cluster Head

Fig 3.6: State Diagram of the Entire System

3.7 Summary

This chapter discussed the high level design, design consideration, system architecture, module specification with the data flow diagram for each module of the Wireless sensor network.

Chapter 4

DETAILED DESIGN

Detailed design is the process of defining the components, modules, interfaces and data for a system to satisfy specified requirements. System model is a phase where an internal logic of each of these modules specified in high level design is decided. In this phase further details and algorithmic design of each of these modules is specified. Other low-level components and subcomponents are also described as well. This chapter also discuss about the control flow in the software with much more detail about software modules by clarifying the details about each function with functionality, purpose, input and output. Detailed design of each module of our project is as described below.

4.1 Flow Chart

A flow chart is a type of diagram that represent algorithm, work flow or process, showing the steps of boxes of various kinds, and there order by connecting them with arrows. A flow chart is a common type of chart, which represents an algorithm or process, showing the steps as boxes of various kinds, and their order by connecting these with arrows.

The diagrammatic representation illustrates a solution model to a given model, flow charts are used in analyzing, designing, documenting or managing a process or program in various fields. Flow chart helps in visualize what is going on and thereby help understand a process, and perhaps also find flaws, bottle necks and other less-obvious features in it. There are many different types of flow charts, and each type as its own repertoire of boxes and national conventions. The two most common types of boxes in a flow chart are a process step, usually called activity, and denoted as rectangular box and a decision usually denoted as a diamond. It is used to develop understanding of how a process is done, to study a process for improvement, communicate to others how a process is done, when better communication is needed between people involved with a same process to document a process, when planning a project. It defines the process to be diagrammed. The boundaries of the process is discussed and decided where or when does the process starts and ends. The level of detail to be included is the diagram. The activities are arranged in proper sequence.

4.1.1 Flowchart for Grid Head Election

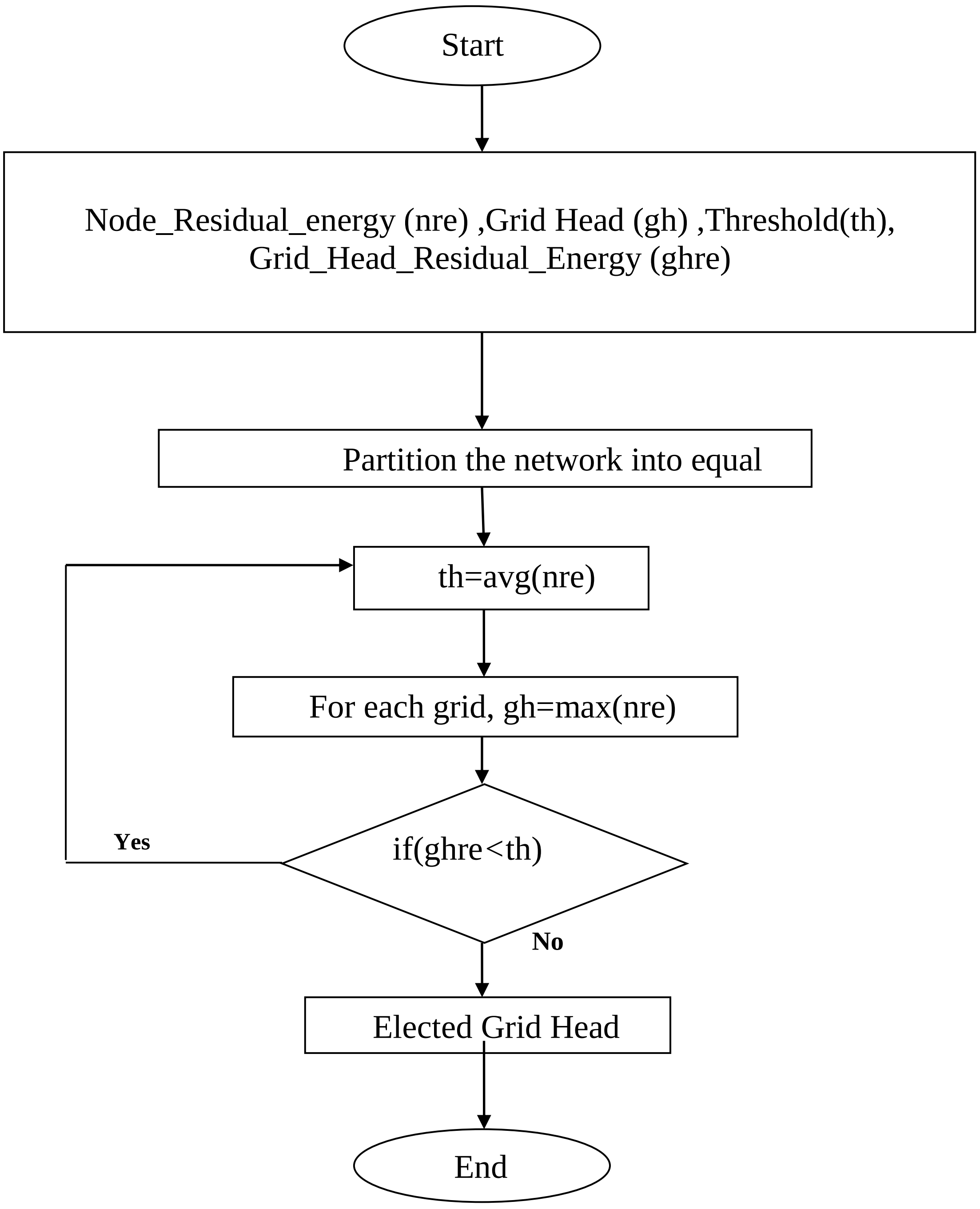
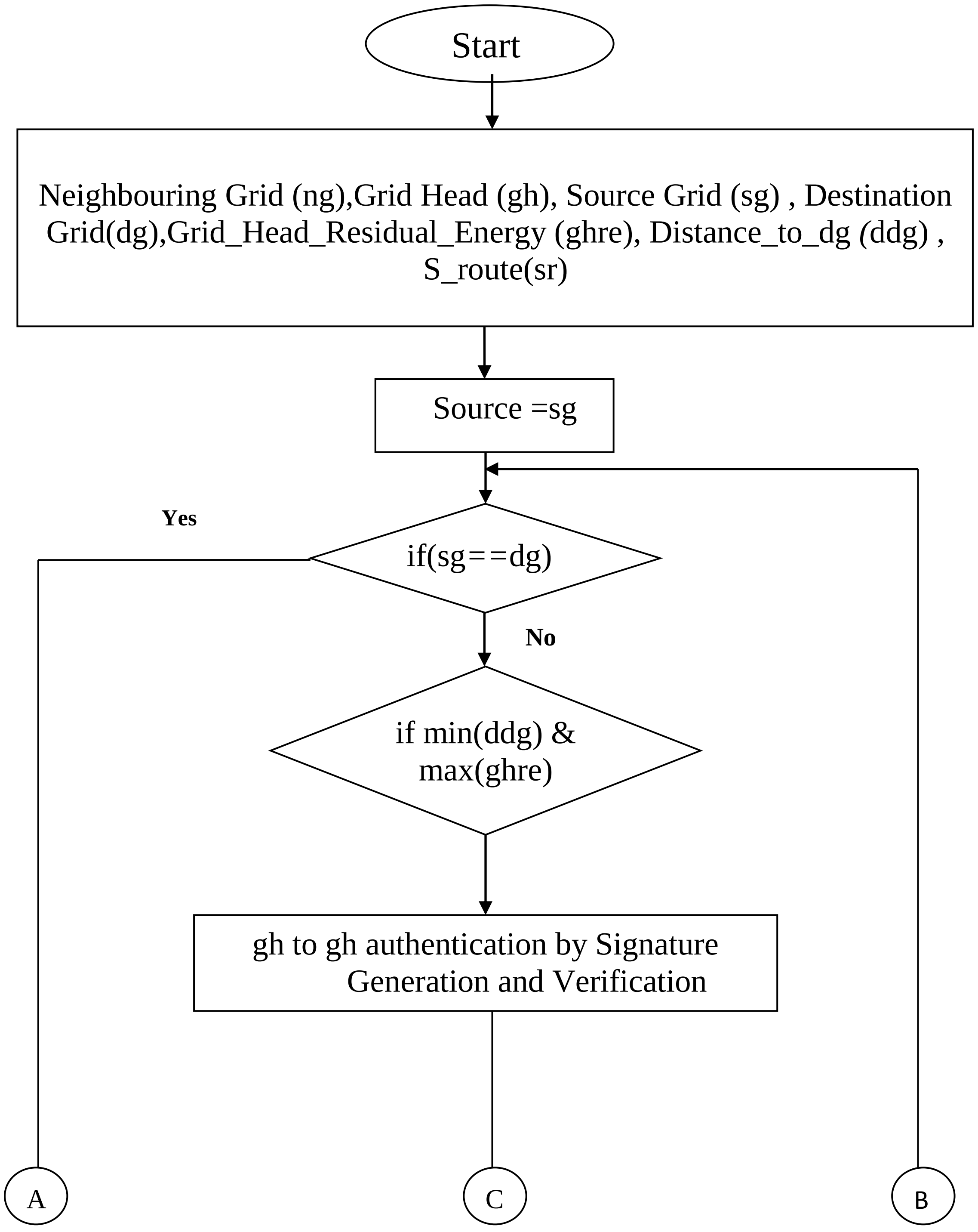

Start
Node_Residual_energy (nre) ,Grid Head (gh) ,Threshold(th),
Grid_Head_Residual_Energy (ghre)
Partition the network into equal
th=avg(nre)
For each grid, gh=max(nre)
if(ghre<th)
Yes
No
Elected Grid Head
End

Fig.4.1: flowchart of grid head election.

The Fig 4.1 shows the process of election of grid head. The network which consists of n number of nodes is partitioned into equal sized grids. The residual energy of each nodes are calculated. Initially the residual energy of all the nodes is set to 10 and after some iterations energy of few nodes goes down. The average node residual energy of all the nodes have been set as threshold. For each grid, the grid head is selected based on the maximum node residual energy. If grid head residual energy is lesser than the threshold value, then again new threshold value is assigned for the next average node residual energy and the process continuous like before and again it checks whether the grid head residual energy is lesser than the threshold value and if the grid head residual energy value is greater than that of the threshold value, then that grid head is selected.

4.1.2 Flowchart for Shortest Route Discovery


Start
Neighbouring Grid (ng),Grid Head (gh), Source Grid (sg) , Destination Grid(dg),Grid_Head_Residual_Energy (ghre), Distance_to_dg (ddg) , S_route(sr)
Source =sg
Yes
if(sg==dg)
No
if min(ddg) & max(ghre)
gh to gh authentication by Signature Generation and Verification
A
C
B

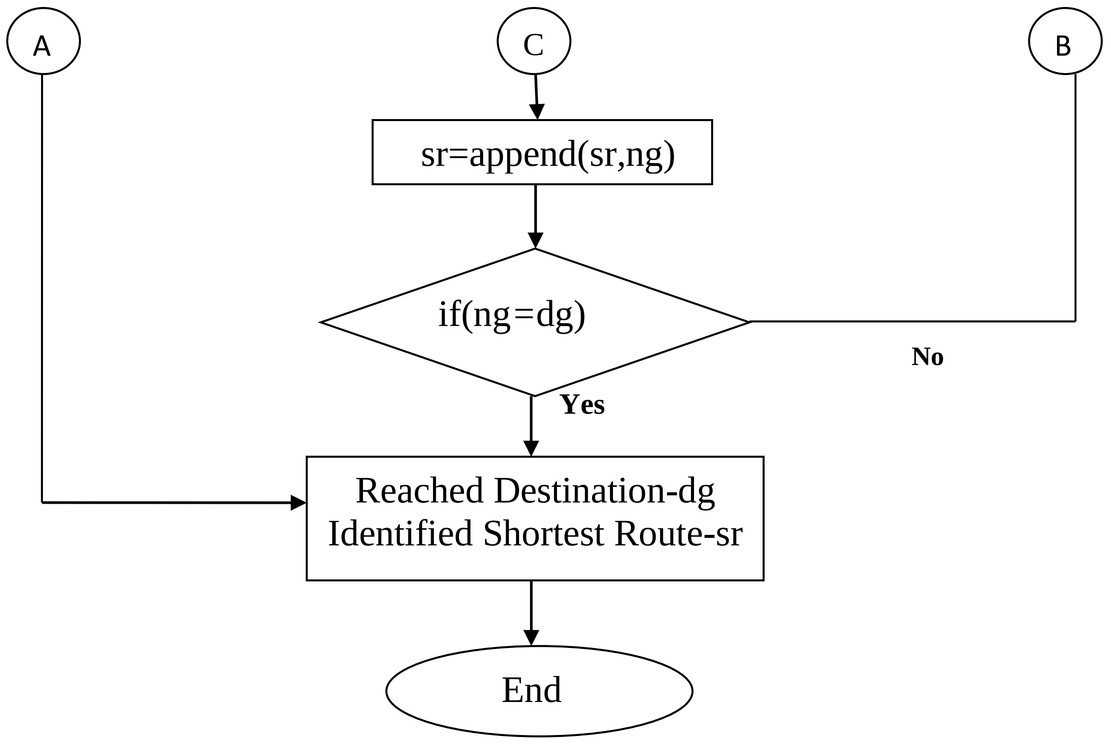


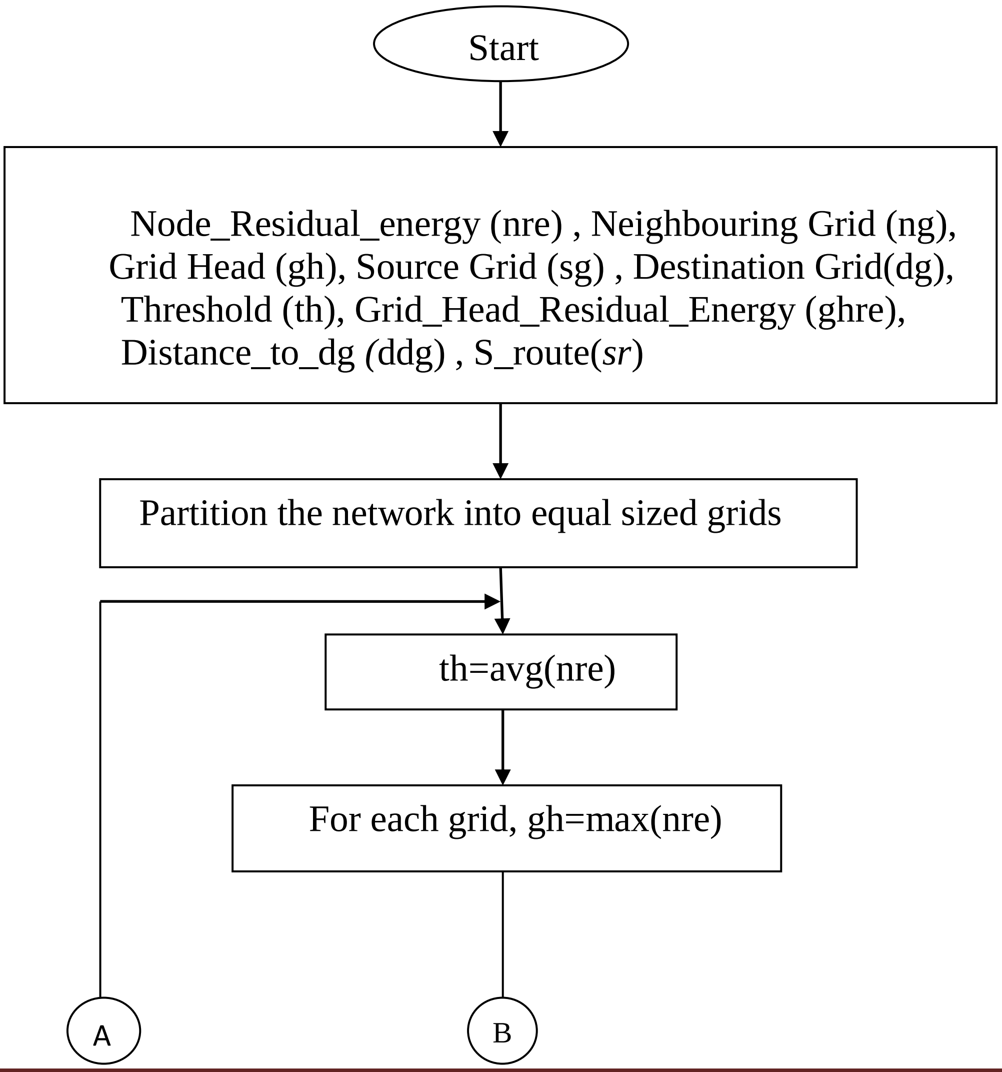
Fig 4.2: flowchart of Shortest Route Discovery.

The Fig 4.2 shows the flowchart of shortest route discovery. To find the shortest route here we assign source as source grid(sg). If source grid is equal to destination grid , which forms a self loop and thus the destination is reached and shortest route is identified. If source is not equal to destination grid then we compute minimum distance to the destination grid and also find the node which have maximum grid head residual energy. Then the grid head to grid head authentication and verification is processed. The shortest route gets updated by adding the neighboring grid to the shortest route and if the neighboring grid equals the destination grid then the destination is reached and it is identified as the shortest route. If the neighboring grid does not equals destination grid then again the selection of source takes place.

4.1.3 Flowchart for Entire System

The Fig 4.3 represents the overall flowchart which is divided into two sections in which one is for grid head election and another for shortest route discovery. All the components with their nomenclatures are given as an input at the beginning. The process of grid head election involves the partition of networks into equal sized grids. Threshold (th) is assigned for average node residual energy of all the nodes. For each grid, grid head equals maximum value of node residual energy (nre) . If the grid head residual energy (ghre) is more than that of threshold (th) then it is elected as the grid head, otherwise again new average node residual energy is assigned as threshold then the same process continues and repeats till the grid head residual energy is greater than or equal to the threshold.

In the next section, to find the shortest route here we assign source as source grid(sg). If source grid is equal to destination grid , which forms a self loop and thus the destination is reached and shortest route is identified. If source is not equal to destination grid then we compute minimum distance to the destination grid and also find the node which have maximum grid head residual energy. Then the grid head to grid head authentication and verification is processed. The shortest route gets updated by adding the neighboring grid to the shortest route and if the neighboring grid equals the destination grid then the destination is reached and it is identified as the shortest route. If the neighboring grid does not equals destination grid then again the selection of source takes place.


Start
Node_Residual_energy (nre) , Neighbouring Grid (ng), Grid Head (gh), Source Grid (sg) , Destination Grid(dg), Threshold (th), Grid_Head_Residual_Energy (ghre), Distance_to_dg (ddg) , S_route(sr)
Partition the network into equal sized grids
th=avg(nre)
For each grid, gh=max(nre)
A
B

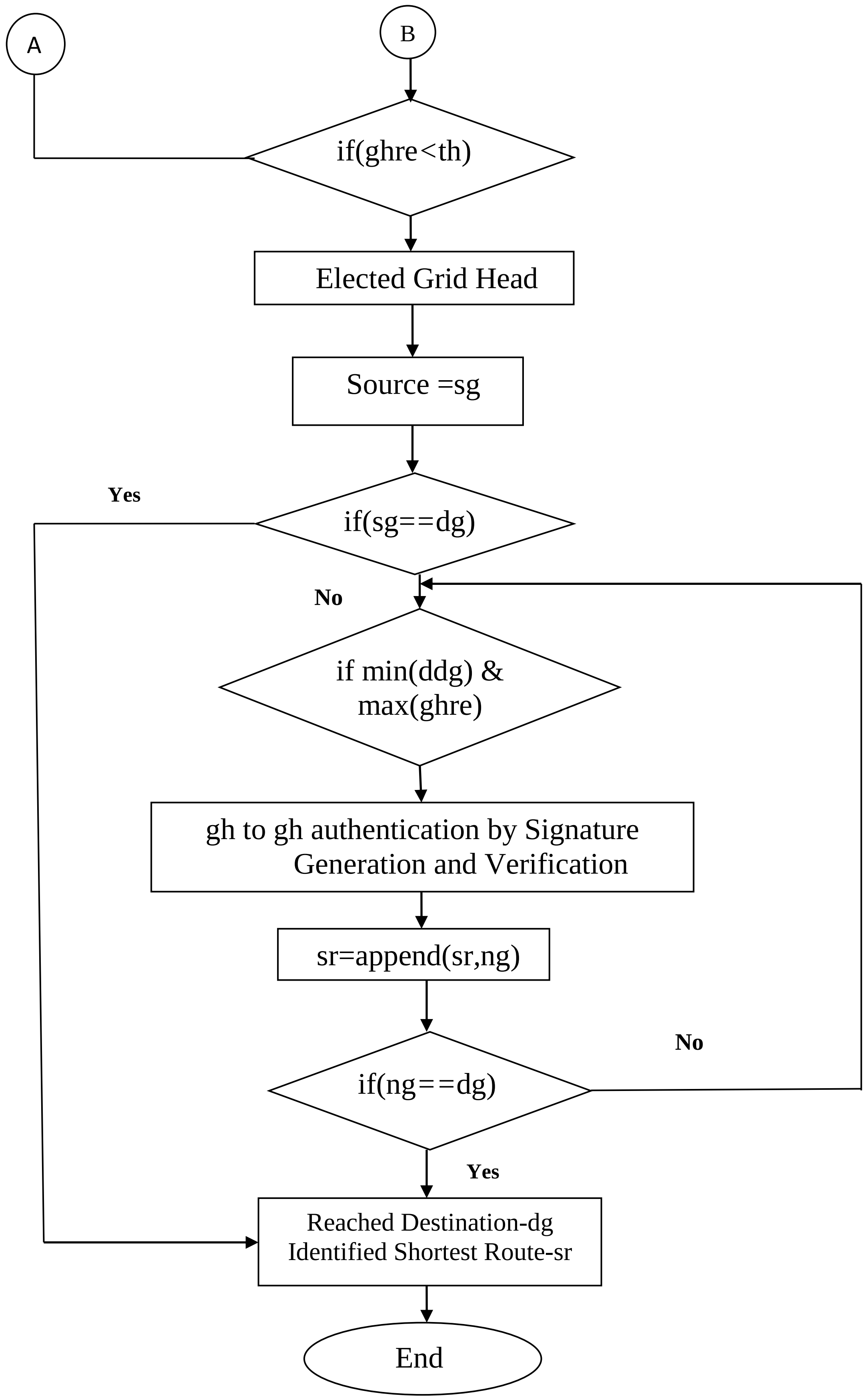

A
B
if(ghre<th)
Elected Grid Head
Source =sg
Yes
if(sg==dg)
No
if min(ddg) & max(ghre)
gh to gh authentication by Signature Generation and Verification
sr=append(sr,ng)
No
if(ng==dg)
Yes
Reached Destination-dg Identified Shortest Route-sr
End

Fig 4.3: Flowchart for Entire System

4.2 Summary

This chapter discussed the detailed level design, detailed description of Wireless sensor network that includes flowchart which shows the actual flow of the system. This chapter includes flowchart of grid head election, flowchart of shortest route discovery and flowchart for the entire system.

Chapter 5
IMPLEMENTATION

The implementation phase of any project development is the most important phase as it yields the final solution, which solves the problem at hand. The implementation phase involves the actual materialization of the ideas, which are expressed in the analysis document and developed in the design phase. Implementation should be perfect mapping of the design document in a suitable programming language in order to achieve the necessary final product. Often the product is ruined due to incorrect programming language chosen for implementation or unsuitable method of programming. It is better for the coding phase to be directly linked to the design phase in the sense if the design is in terms of object oriented terms then implementation should be preferably carried out in a object oriented way. The factors concerning the programming language and platform chosen are described in the next couple of sections.

The implementation stage in a system project involves:

- Careful planning
- Investigation of the current system and the constraints on implementation
- Training of staff in the newly developed system.

5.1 Programming Language Used

Java is a general-purpose computer-programming language that is concurrent, class-based, object-oriented and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere” meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to byte code that can run on any Java virtual machine (JVM) regardless of computer architecture. As of 2016, Java is one of the most popular programming languages in use particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun relicensed most of its Java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java (bytecode compiler), GNU Class path (standard libraries), and IcedTea-Web (browser plugin for applets).

5.2 Algorithm for Resource Constraint Secure Routing Protocol

The Pseudo code used for different module development of the project are defined in the following sections.

5.2.1 Pseudocode for adding nodes from the client side and refreshing the network

```
Step1: Enter the IP address of the destined server.
        jLabel1.setText("IP Address");
Step2: Click on the connect button.
        jButton1.setText("Connect");
Step3: A node will be added at the server.
        jButton1.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt)
        jButton1ActionPerformed(evt); }
Step4: A node id will be generated for each and every node that will be added into the
        Network.
        jLabel2.setText("Node Id");
Step5: Click on the Refresh network button inorder to upadate and display the nodes
        being added into the network.
        jButton5.setText("Refresh Network Information");
        jButton5.setEnabled(false);
        jButton5.addActionListener(new java.awt.event.ActionListener() {
        actionPerformed(java.awt.event.ActionEvent evt)
        jButton5ActionPerformed(evt); }
Step6: Create an interface in between client and server.
```

```
        CommunicationInterface = (CommunicationInterface) Naming
        .lookup("//" + jTextPane1.getText()+"/RmiServer");
```

5.2.2 Pseudo code for manually adding the nodes into the network

```
Step1: Click on the manual network configuration in the server system.
Step2: Enter the number of nodes to be added.
Step3: Establish a desired network using the matrix.
Step4: Click on the submit button
Step5: Network consisting of the desired number of nodes will be created.
        getManualNodeConfiguration()
        JButton addNode = new JButton("Manual Network Configuration");
        addNode.addActionListener(new ActionListener()
       TaskTable frame = new TaskTable(null,network,n);
       frame.setTitle("Manual Network Configuration");
       buttonPanel.add(frame.getSubmitButton());
       frame.add(buttonPanel);
```

5.2.3 Pseudo code for sending, forwarding and receiving the packets

```
Step1: Click on the add node button in order to add any number of nodes in the network.
        AddNewNode()
        (network! = null)
        Then add a new node.
Step2: Select the source, the destination and click on the send button.
        mFrom = "" + from;
        mTo = "" + to;
        data = message;
        packets = Packet.SendPackets(message, ""+from, ""+to);
        So that the message packets will be sent from the source node to its destination node.
Step3: The packet is sent from a particular node and received by a particular node after being forwarded by one or ‘n’ number of nodes.
            Packet ForwardPacket(Packet p, String currentNode, String nexNode)
             Packet np = new Packet();// new packet being forwarded
```

```
np.From = currentNode;//from a cuurent node
np.To = nextNode;//to a next node
np.Source = p.Source;//source node
np.Destination = p.Destination;//destination node
np.message = p.message;
np.Time = ""+System.currentTimeMillis();
np.Type = "F";//Forwarded
Packet ReceivePacket(Packet p, String currentNode)
Packet np = new Packet();// a new packet being received
np.From = p.From;//from a source node
np.To = currentNode;//to a current node
np.Source = p.Source;//
np.Destination = p.Destination;
np.message = p.message;
np.Time = ""+System.currentTimeMillis();
np.Type = "R";// Received
```

5.2.4 Pseudo code for finding the shortest path

```
Step1: Activate the proposed system
Step2: Click on the Elect Cluster head button.
Step3: Click on the form grid button.
     JOptionPane.showMessageDialog(null, "Elected Cluster Heads are"+
    gridheads.toString());
Step4: Click on the send button in order to send the packets
The packets will be sent from the source to the destination through the shortest path.
     currpath = new LinkedList<String>();
     new DijkstraShortestPath(mGraph);
      alg.getPath(mFrom, mTo);
       jpPathType.add(getElectClusterHead();
       jpPathType.add(getFormGrids();
       String source = mGraph.getSource(c);
       String dest = mGraph.getDest(c);
       path.addEdge(source + dest, source, dest);
```

```
        return shortestpath
```

5.2.5 Pseudo code for updating a node’s battery level

```
Step1: Select the source and the destination.
Step2: Send the packets.
Step3: Click on the battery button in order to check out the battery level of each node.
    newbatterylevel =batteryLevel.get(v).intValue()-1;
    batteryLevel.put(v, newbatterylevel);
```

5.2.6 Pseudo code for the calculation of average battery level of the nodes in the network

```
Step1: Select the proposed system.
Step2: Send the packets after selecting source and destination.
Step3: Click on the generate report button.
Step4: Open the average battery information file.
    getProposedAverageEnergy(batteryLevel);
    double result = 0.0;
    double total = 0.0;
    for(String s:batteryLevel.keySet())
    total = total + batteryLevel.get(s);
   result = total / batteryLevel.size();
   return result
```

5.2.7 Pseudo code for the generation of report

```
Step1: Add any desired number of the nodes to form a network.
Step2: Select existing system.
Step2: Select a source node.
Step3: Select a destination node.
Step4: Send the message packets.
Step5: Select proposed system.
Step6: Form grid and elect cluster heads.
Step7: Send the message packets.
Step8: Generate report.
```

```
reports = new JButton("Reports");
reports.addActionListener(new ActionListener()
actionPerformed(ActionEvent ae)
DisplayGraph.energyPut();
JOptionPane.showMessageDialog(null, "Graphs Successfully Generated");
```

Step9: Graph of existing system v/s proposed system of the average battery levels are generated .

5.2.8 Pseudo code for displaying the battery level of an existing system and our proposed system

```
Step1: Click on the generate report.
Step2: Open batterylevel.jpeg image
Step3: The battery levels of both the systems after each iteration will be displayed in the form a graph.
barChart = ChartFactory.createLineChart3D(
          "Average Battery Level",
          "Iterations",
          "Battery Level",
           width = 640; //Width of the image
          height = 480; // Height of the image
          File barChart3D = new File( "BatteryLevel.jpeg" );
          ChartUtilities.saveChartAsJPEG( barChart3D, barChart, width, height);
//JPEG image containing comparison of the battery levels of both the systems
         graphically.
```

5.6 Summary

This chapter provides the detailed explanation and description about the programming language used platform selection and finally explains the pseudo code for each implementation modules.

Chapter 6

TESTING

Testing is an important phase in the development life cycle of the product; this was the phase where the error remaining from all the phases was detected. Hence testing performs a very critical role for quality assurance and ensuring the reliability of the software. During the testing, the program to be tested was executed with a set of test cases and the output of the program for the test cases was evaluated to determine whether the program is performing as expected. Errors were found and corrected by using the following testing steps and correction was recorded for future references. Thus, a series of testing was performed on the system before it was ready for implementation.

6.1 Types of Testing

Unit Testing

Individual components are tested to ensure that they operate correctly. Each component is tested independently without other system components. Unit testing and module testing are usually responsibility of the programmers developing the components. Here we are making our own test data and incrementally test the code as it is developed. Unit testing is a part of the implementation process and it is expected that a component conforming to its specification will be delivered as a part of the process.

Integration Testing

Integrated testing is the systematic testing to uncover the errors with an interface. This testing is done with simple data and developed system has run successfully with the simple data. The need for integrated system is to find the overall system performance.

Types of integration testing

- **Top down integration**
  - ➢ Modules are integrated by moving downward through the control hierarchy, beginning with the main program module.
- **Bottom up integration**

  Bottom up integration is implemented with the following step:
  - ➢ Low level modules are combined into clusters that perform a specific

Software sub-function.

- A driver (i.e.,) the control program for testing is written to coordinate test case input and output. The cluster is tested.
- Drivers are removed and clusters are combined moving upward in the program structure.

6.2 Test Cases

TEST CASES	DESCRIPTION	EXPECTED RESULT	OBTAINED RESULT	REMARK Pass or Fail
Case A: Creating a node	Enter the IP address of the destined Server system	It should create a node in the server system	A Node is created	Pass (Snapshot 7.1)
	Enter the IP address of any other system	It should not create a node in the server system	A Node is not created	Pass (Snapshot7. 2)

Case B: Cluster head election	Elect Cluster head	Node with the highest residual energy should be elected	The Highest residual energy Node is elected	Pass (Snapshot 7.5)

Case C: Shortest Paths	Enter Source node number Enter Destination node number Send Message (Existing system)	It should find the shortest path between Source and Destination	Shortest path is not determined as the information travels through all the possible paths and the nodes before reaching the destination node	Pass (Snapshot 7.6)
	Enter Source node number Enter Destination node number Send Message (Proposed system)	It should find the shortest path between Source and Destination	Shortest path is determined	Pass (Snapshot 7.4)

Case D: Find an alternative Cluster head after each iteration	Check the residual energy of each node	Should find an alternative Cluster head if the residual energy of the present cluster head is less than the threshold value	An alternative cluster head i.e. node whose residual energy is higher than the threshold value is elected	Pass (Snapshot 7.8)

Table 6.1: Test Cases.

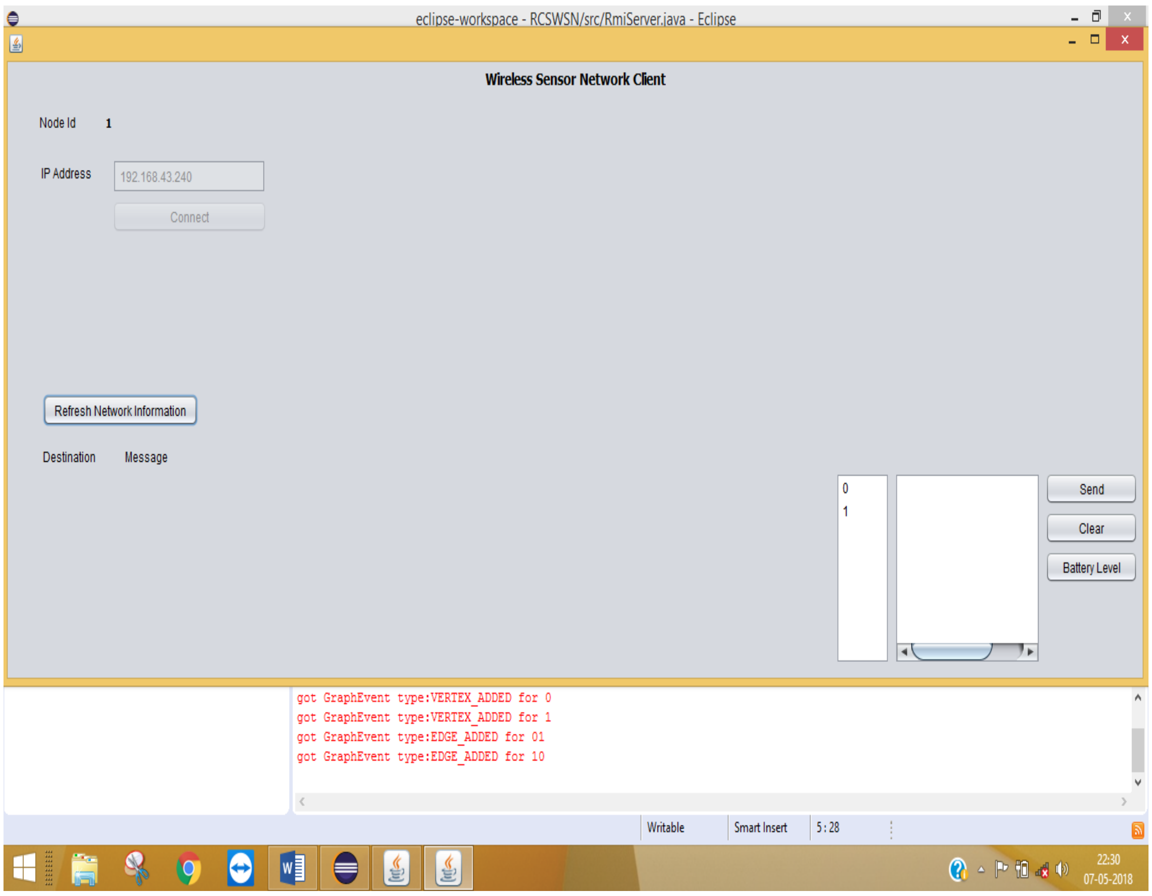
6.3 Summary

This chapter describes System testing. Different cases of testing's are discussed in this chapter.

CHAPTER 7

RESULTS AND DISCUSSIONS

7.1 Snapshots


eclipse-workspace - RCSWSN/src/RmiServer.java - Eclipse
Wireless Sensor Network Client
Connect
Refresh Network Information
Destination
Message
Send
Clear
Battery Level
Writable
Smart Insert
5 : 28
22:30
07-05-2018

Node Id 1

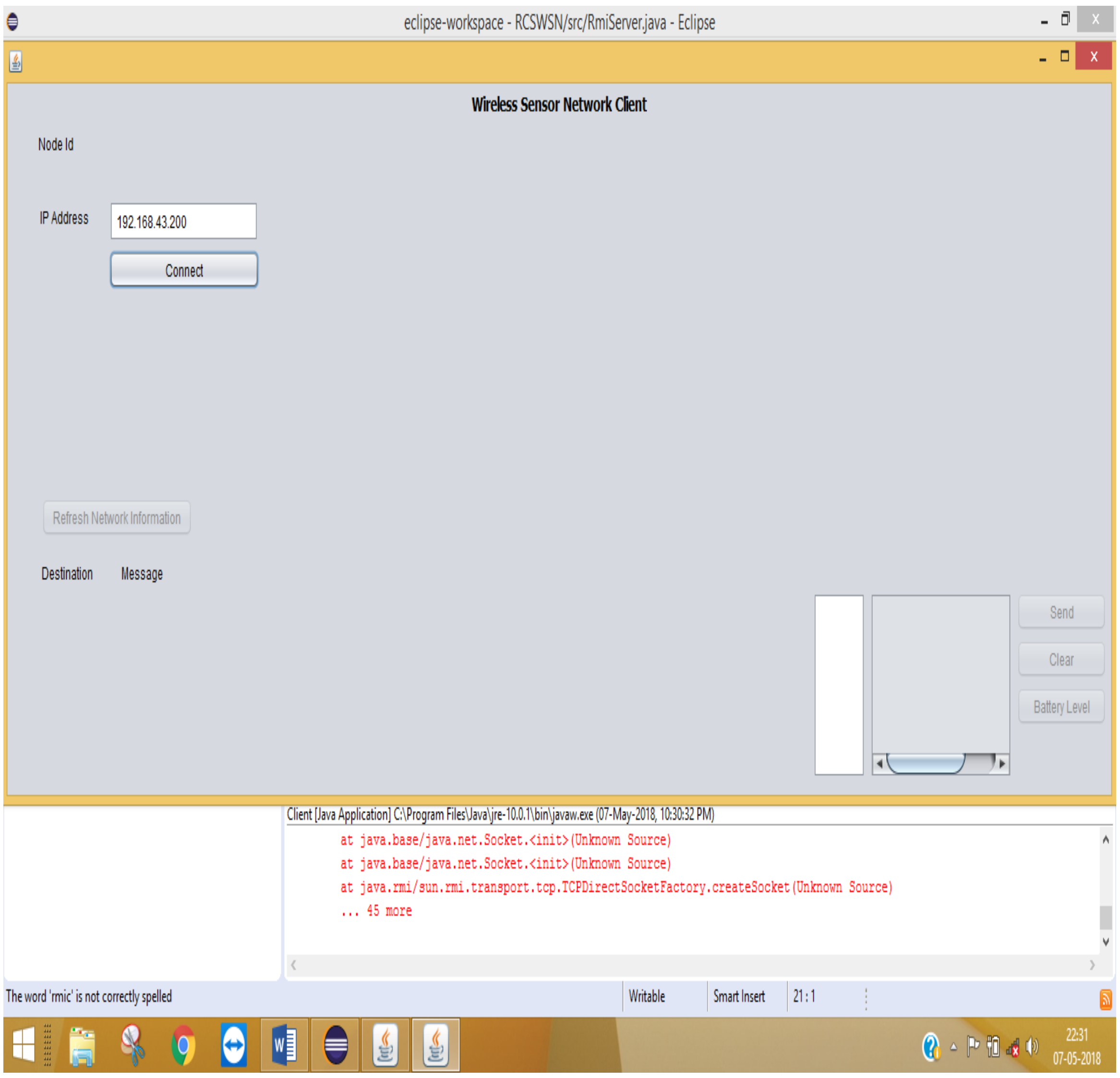
IP Address 192.168.43.240

```
0
1
```

```
got GraphEvent type:VERTEX_ADDED for 0
got GraphEvent type:VERTEX_ADDED for 1
got GraphEvent type:EDGE_ADDED for 01
got GraphEvent type:EDGE_ADDED for 10
```

Snapshot 7.1: To show the creation of nodes.

When the correct IP address of the server is entered in the wireless sensor network client. This shows the creation of nodes through client side that is by entering the ip address of the server. In the client system as the ip address of the server is given as the input and as the client nodes are created and connected in the server they tend to generate some unique node id and generally this unique node id would be any integer in ascending form that is at first when a node is connected to server through client, Initially the node id would be 0 and by connecting again and again in the same way the node id value generated might vary as 1,2,3 and so on . By Refreshing the network information, all the created nodes are displayed which is shown in the Snapshot 7.1.

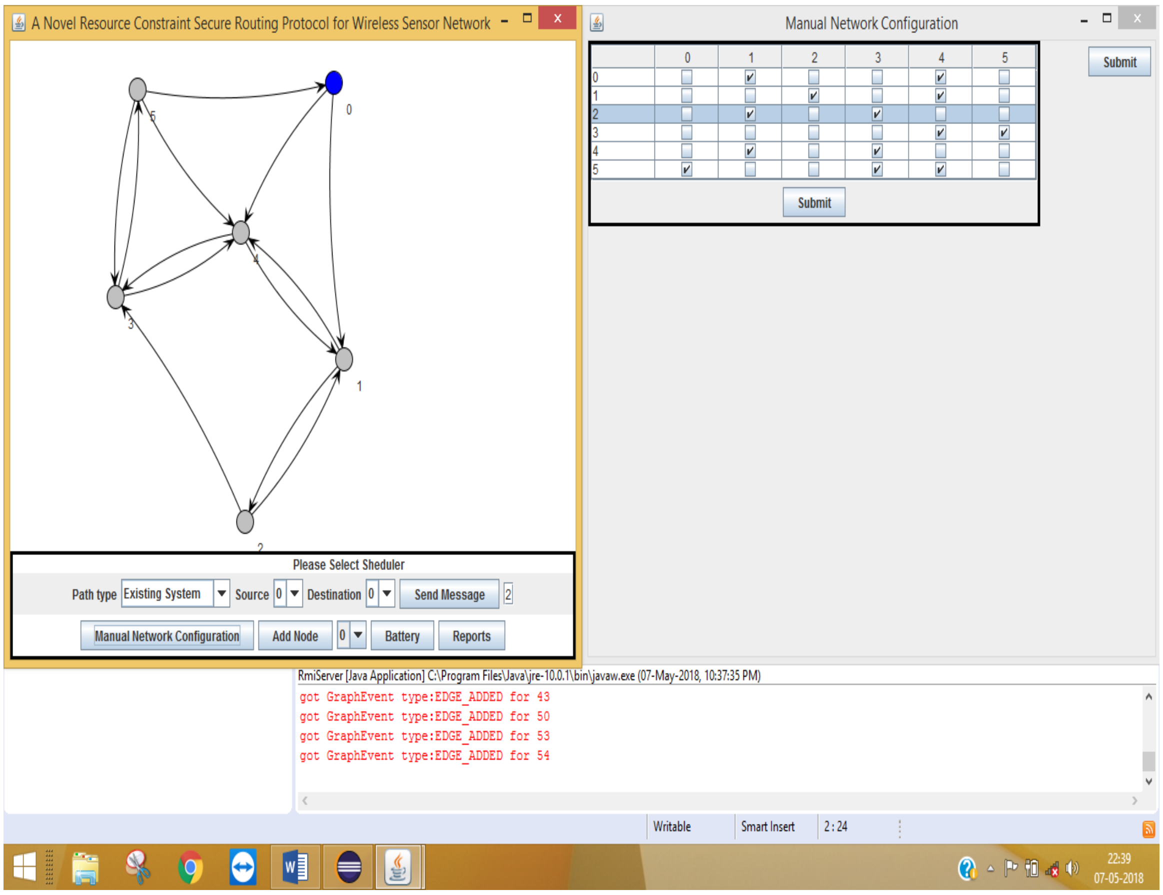

eclipse-workspace - RCSWSN/src/RmiServer.java - Eclipse
Wireless Sensor Network Client
Node Id
IP Address
Connect
Refresh Network Information
Destination
Message
Send
Clear
Battery Level
The word 'rmic' is not correctly spelled
Writable
Smart Insert
21 : 1
22:31
07-05-2018

IP Address: 192.168.43.200

```
Client [Java Application] C:\Program Files\Java\jre-10.0.1\bin\javaw.exe (07-May-2018, 10:30:32 PM)
        at java.base/java.net.Socket.<init>(Unknown Source)
        at java.base/java.net.Socket.<init>(Unknown Source)
        at java.rmi/sun.rmi.transport.tcp.TCPDirectSocketFactory.createSocket(Unknown Source)
        ... 45 more
```

Snapshot 7.2: To show an Unknown source.

In the client side we need to give input as the servers IP address to establish the clients connection in to the server. But one thing is to be taken care of that is the entered IP address of the server is correct or not. If the correct IP address of the server is entered in the clients system, then proper connection of that client with the server is made and in the same manner many clients are connected to the server. But when a client system enters the wrong IP address that is the IP address would be something and the client system enters something else that is IP address of some other system which might not be a server at all, In these sort of cases the only thing happens is that the connection would get failed and the client would not be connected to any server and nodes cannot be created of that client. This is shown in the Snapshot 7.2.

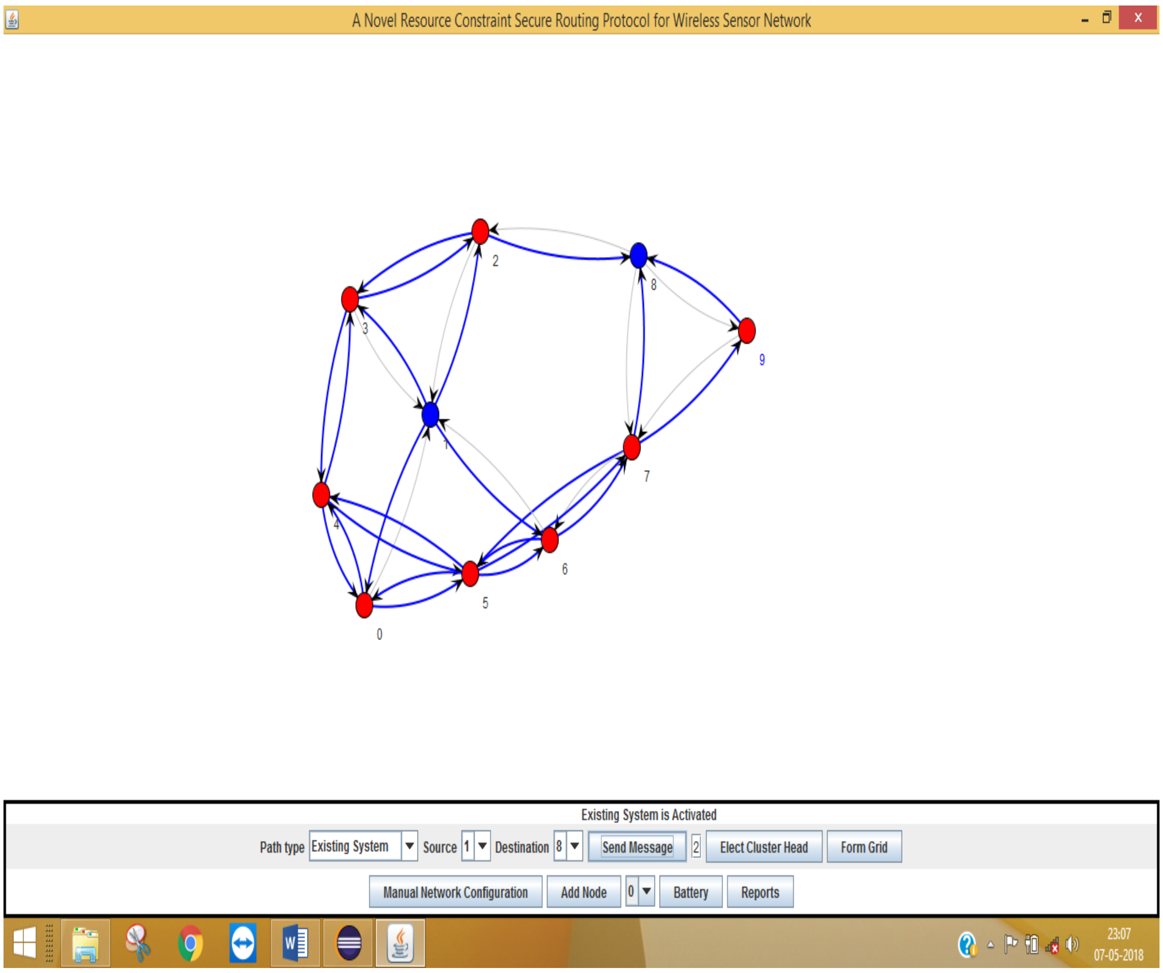


	0	1	2	3	4	5
0	☐	✓	☐	☐	✓	☐
1	☐	☐	✓	☐	✓	☐
2	☐	✓	☐	✓	☐	☐
3	☐	☐	☐	☐	✓	✓
4	☐	✓	☐	✓	☐	☐
5	✓	☐	☐	✓	✓	☐

```
RmiServer [Java Application] C:\Program Files\Java\jre-10.0.1\bin\javaw.exe (07-May-2018, 10:37:35 PM)
got GraphEvent type:EDGE_ADDED for 43
got GraphEvent type:EDGE_ADDED for 50
got GraphEvent type:EDGE_ADDED for 53
got GraphEvent type:EDGE_ADDED for 54
```

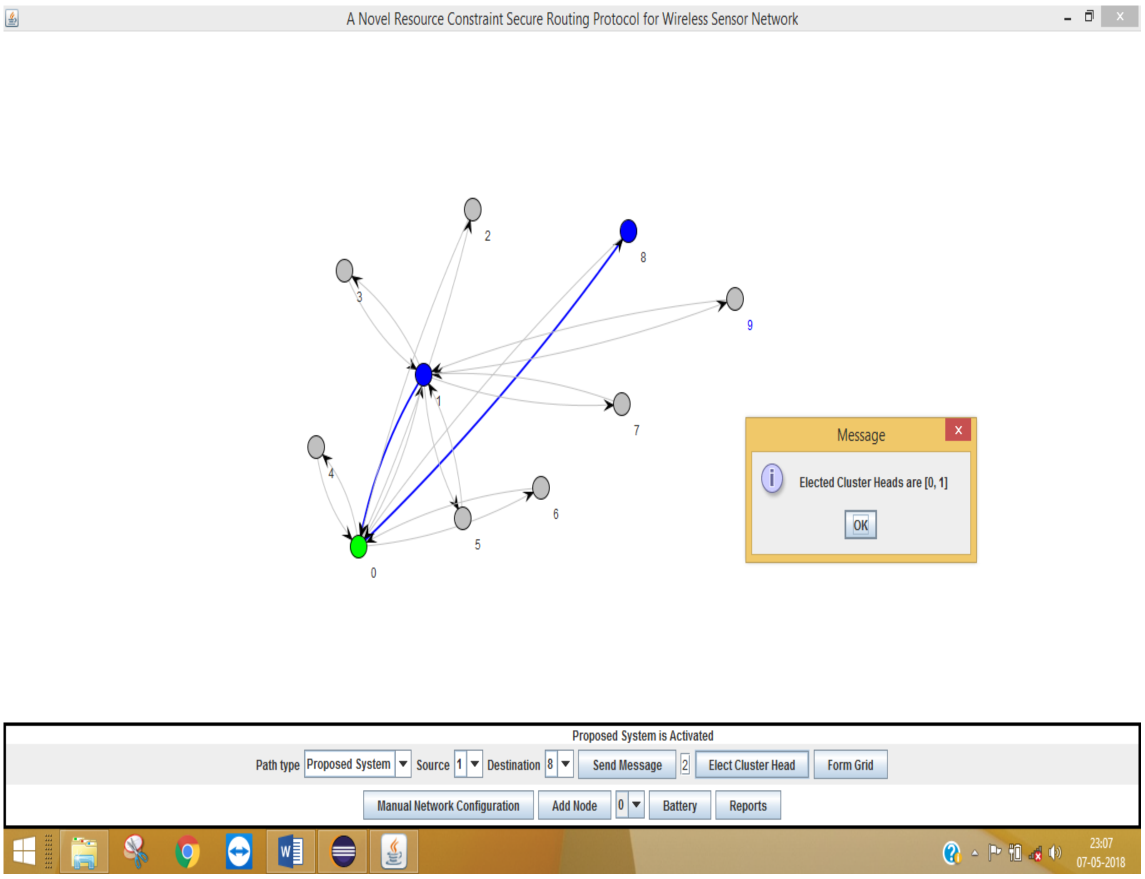
Snapshot 7.3: Manual network configuration.

The snapshot 7.3 shows the manual network configuration. There are few methods used in creating nodes and connecting them to the server, one such method is entering the IP address of the server at the client side and connecting and one more method is adding the nodes directly on the server, there would be some functionality and through which we can add nodes in a faster manner and we can add many number of nodes. Using the above two mentioned methods, nodes can be successfully created and added into the server. This connectivity of several nodes has taken place in random order that is there is no proper order for the connection. It is a random connectivity of nodes and all the nodes are connected in multidirectional manner. If we want to design the connection of nodes to happen in a certain possible way then this can also be done by doing the manual network configurations. In manual network configurations the nodes can be arranged and connected in matrix representation. The manual network configuration is one such method which establishes the appropriate connection between nodes. Which is shown in the Snapshot 7.3.


A Novel Resource Constraint Secure Routing Protocol for Wireless Sensor Network
0
1
2
3
4
5
6
7
8
9
Existing System is Activated
Path type
Existing System
Source
1
Destination
8
Send Message
2
Elect Cluster Head
Form Grid
Manual Network Configuration
Add Node
0
Battery
Reports
23:07
07-05-2018

Snapshot 7.4: Paths taken by a packet in an existing system from source to destination.

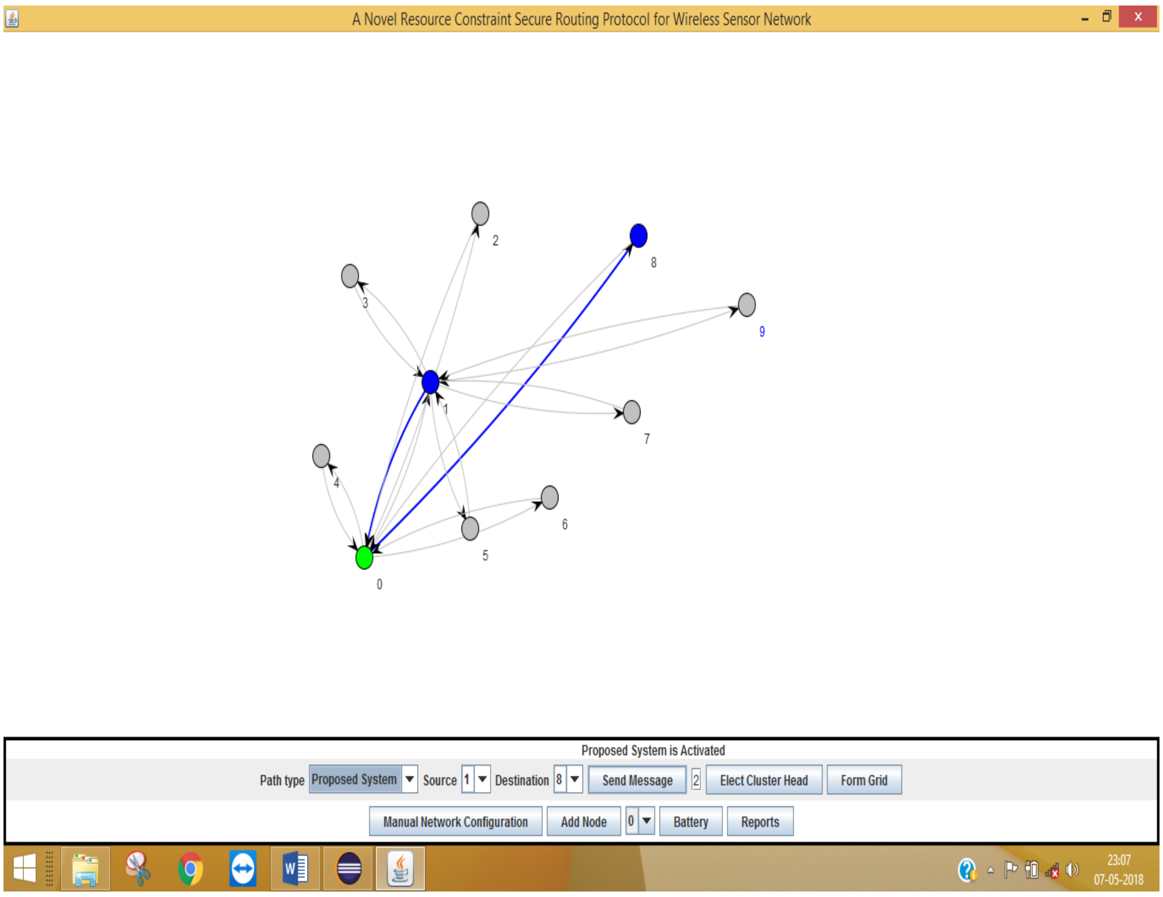
The Snapshot 7.4 shows the paths taken by a message packet which transfers from source to destination through various intermediate nodes. In the existing system after adding the nodes source node and the destination nodes are added to make the transfer of data possible. This mode of transfer of data the data packets are transferred in some route. Even in the existing system we can add nodes directly into the system by pressing the add nodes key for several number of times also we can construct the nodes and associated connection among them by the manual network configurations. By using the manual network configuration we can manually assign the connectivity among different sets of nodes which will be given in matrix form. As and when we select the source and destination nodes, the source and destination nodes do get highlighted and the remaining intermediated nodes are represented by specific color. In the existing system the transfer of data does take place involving every node which possibly comes near the route i.e. it takes all the possible routes.



Elected Cluster Heads are [0, 1]

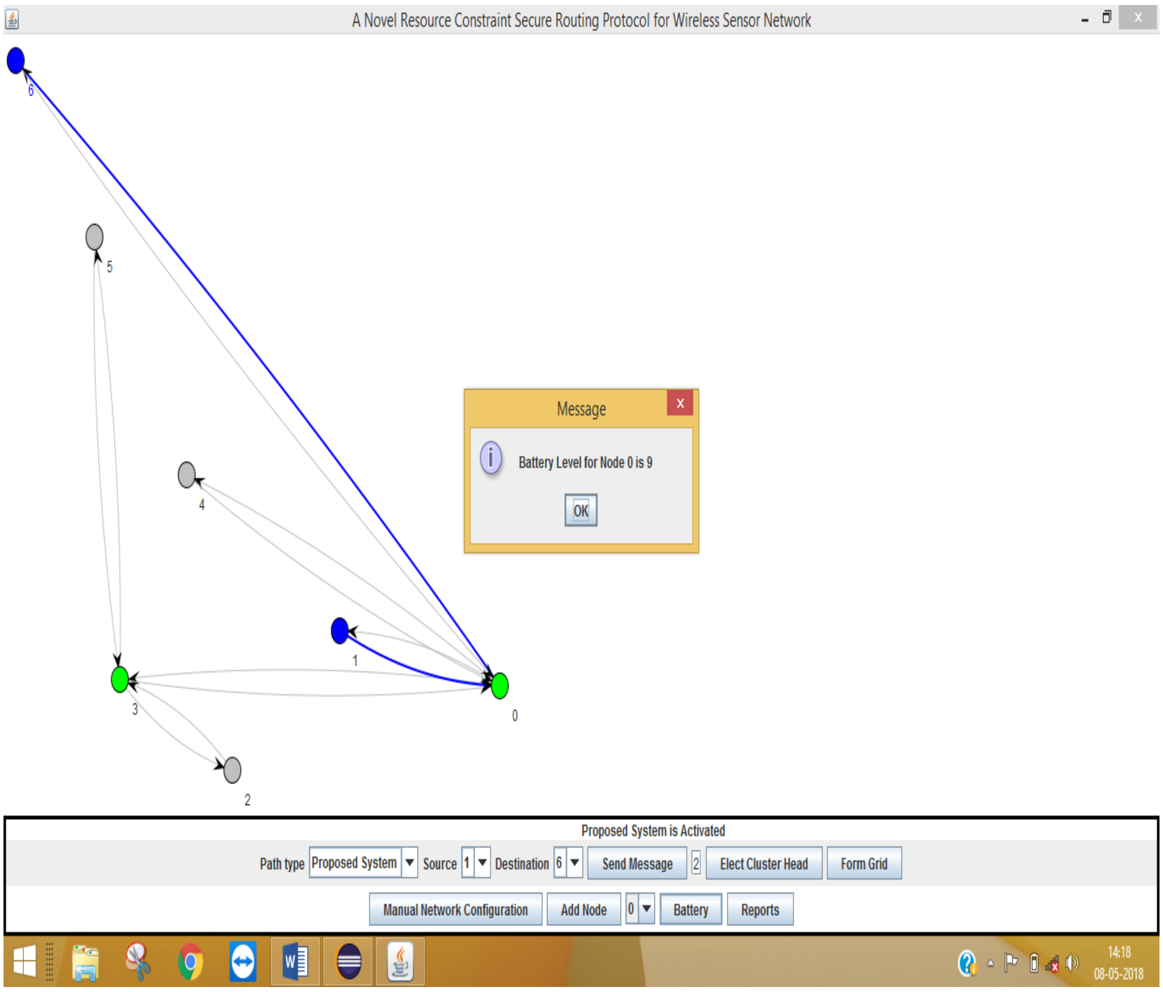
Snapshot 7.5: Election of the Cluster heads.

The Snapshot 7.5 shows the election of cluster head. Usually the election of cluster head takes place in the proposed system. In this system, as the nodes are added into the server. The connection among them is established simultaneously and once the connection of nodes is done the source node and destination node is also selected for the transfer of data to take place. But before selecting the send message key we should proceed by electing the cluster head. The process of election of cluster head involves random selection method in which cluster heads are elected randomly and one more additional functionality is that we can also give the number of cluster head as inputs. If we give 3 and press the elect cluster head key, then 3 random nodes are elected as cluster heads. After the election of cluster head we need to form grid and the formation of grid depends on the number of cluster heads. Then the send message key is pressed and the data flows can be visualized as, the data packets from the source node is sent to one of cluster head and from there the data is passed on to another cluster head and from that cluster head the data is finally passed on to the destination node. In this way the transfer of data flow can be seen in this system.


A Novel Resource Constraint Secure Routing Protocol for Wireless Sensor Network
0
1
2
3
4
5
6
7
8
9
Proposed System is Activated
Path type
Proposed System
Source
1
Destination
8
Send Message
2
Elect Cluster Head
Form Grid
Manual Network Configuration
Add Node
0
Battery
Reports
23:07
07-05-2018

Snapshot 7.6: Paths taken by a packet in the proposed system from source to its destination.

The Snapshot 7.6 shows the paths taken by a message packet from its source to the destination after activating the proposed system. In this system after selecting the source node as 1 and destination node as 8 followed by electing 2 cluster heads and in which the randomly elected cluster heads are sending the message packet from source to destination node. The message packet from the source node is forwarded to the cluster head of the first grid which is node 1, in this case the source node and the elected cluster heads are the same and due to which the source node that is node 1 transfers the data to node 0 which is the elected cluster head of the other grid. Thus the residual energy is saved since the message packets doesn't use all the intermediate nodes and all the paths for their transfer of data from the source to the destination. Perhaps they make sure that the data is transferred from source node to the nearest cluster head with high residual energy and from that cluster head to cluster head of another grid and from there the data is reached at the destination node. It is made sure that the paths taken that is the path in which the data transfer take place is most appropriate one .


A Novel Resource Constraint Secure Routing Protocol for Wireless Sensor Network
6
5
4
1
0
3
2
Proposed System is Activated
Path type
Proposed System
Source
1
Destination
6
Send Message
2
Elect Cluster Head
Form Grid
Manual Network Configuration
Add Node
0
Battery
Reports
14:18
08-05-2018

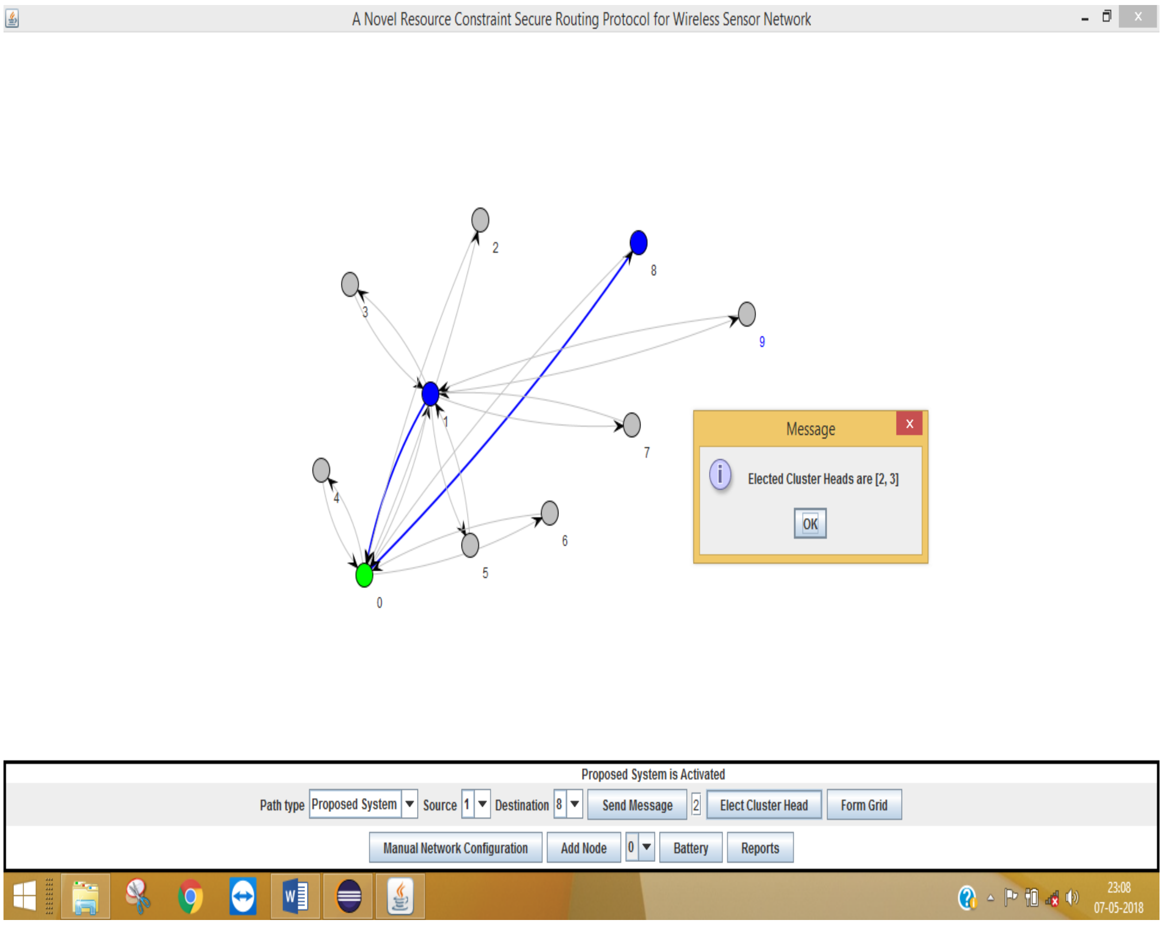
Message

Battery Level for Node 0 is 9

OK

Snapshot 7.7: Displaying the residual energy of node 0.

The Snapshot 7.7 displays the residual energy of a node i.e. node 0. By selecting any node and the battery option, the residual energy of the respective node is displayed. Initially the residual energy of all the nodes in the network will be 10. Like if there are for example 20 nodes in the network the residual energy is assumed as 10 initially and at first the residual energy of all the nodes is said to be 10. Even after the first iteration if the residual energy of each and every nodes are checked it can be analyzed that the residual energy still remain to be 10. So even after the first iteration, the residual energy remains the same i.e. 10 and the residual energy tend to change after several number of iterations. After the first iteration only the residual energy do tend to decrease and it decreases also. But as and when more and more iterations take place the residual energy that is the battery level of each node does really go down it decreases by some value. But using the proposed system and as by comparing the results obtained on the proposed system and existing system, it can be observed that the residual energy of nodes are saved in a very good extent.


A Novel Resource Constraint Secure Routing Protocol for Wireless Sensor Network

Message

Elected Cluster Heads are [2, 3]

OK

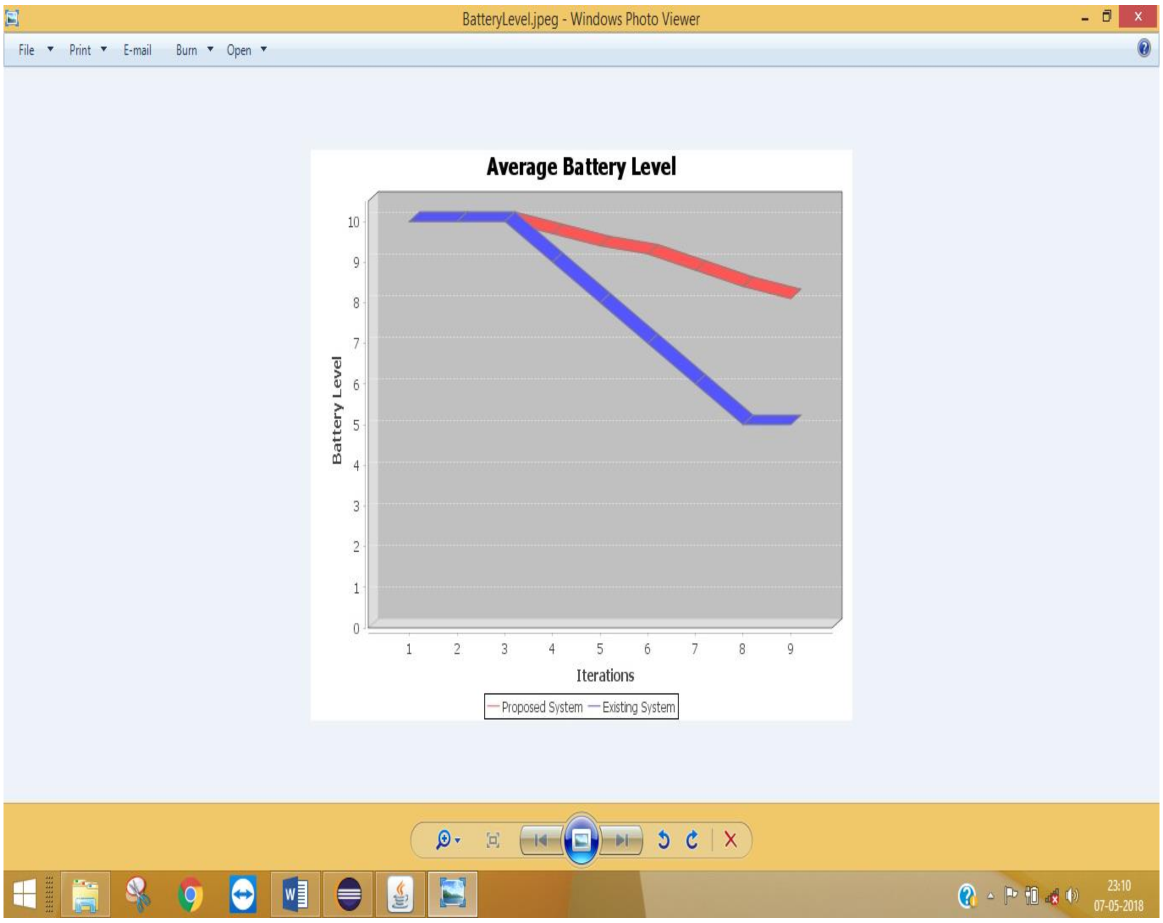
Proposed System is Activated

Path type: Proposed System; Source: 1; Destination: 8; Send Message; 2; Elect Cluster Head; Form Grid

Manual Network Configuration; Add Node; 0; Battery; Reports

Snapshot 7.8: Election of the alternative Cluster heads after few iterations.

The Snapshot 7.8 shows the election of alternative cluster heads after each iteration of sending a message packet. Initially the cluster heads will be node 0 and node 1 as shown in the Snapshot 7.7,after some iterations node 2 and node 3 are elected as the cluster heads as their residual energy is more than the threshold value and the residual energy of node 0 and node 1 is no more higher than the threshold value after few iterations. So the election of cluster head is directly dependent upon the residual energy of those nodes and by pressing the battery key on the server screen the battery life or the residual energy of each and every nodes can be determined. By pressing the elect cluster head key, the cluster heads are randomly assigned and it does assign based on the criteria of higher residual energy. The user can give any number for the cluster heads to be applied i.e. if the cluster heads are given as 4 then from source to destination nodes the data does get transferred from 4 cluster heads and the formation of grid does depend on the number of cluster heads. Generally if there are more set of nodes in the network, then more number of cluster heads would be elected.



Snapshot 7.9: Average battery level of existing system v/s proposed system.

The Snapshot 7.9 is showing the comparison of the average residual energy in between the existing system and the proposed system. The residual energy of the nodes present in the existing system is decreasing rapidly after each iteration of the message packets being sent, whereas the residual energy consumption is reduced considerably in the proposed system.

7.2 Summary

In this results are discussed through snapshots which shows the process involved in creating the nodes, paths taken by the packets from source to the destination and the average residual energy information.

Chapter 8

CONCLUSION AND FUTURE ENHANCEMENT

8.1 Conclusion

Energy efficiency is one of the major problems in wireless sensor network as many nodes tend to consume more energy and preserving the energy level of nodes in some proportion would be considered as an challenging task. The proposed work aims at reducing the consumption of residual energy of each nodes and transferring the data packets in shortest route possible.

This Scheme involves geography based efficient RCS routing protocol for WSNs without relying on flooding. The lifetime optimization of the nodes in the network has been maximized using the proposed routing protocol. RCS combines the technique of deterministic shortest path algorithm with high energy balance ratio. Since the proposed routing protocol does not use random walking technique, the probability of using the low energy node as relay node decreases. Thus, lifetime of the whole network is increased by the routing protocol which provides high energy optimization, high hop-by-hop authentication of messages.

8.2 Future Enhancement

As the concept is all about saving the residual energy of several nodes and giving more life to the nodes. The transfer of data is generally secured but further more security can be enhanced in future which makes the data encryption to a greater extent.

The proposed system is more concerned of saving the residual energy i.e. battery life of nodes in every way possible. Its more about saving energy of nodes and still the data transfer takes place in shortest route possible. But more appropriate paths for transfer of data in faster way can be done as future enhancement.

REFERENCES

[1] R. GEETHA, E. KANNAN in (2017). A novel Resource Constraint Secure routing protocol for wireless sensor networks. IEEE Network.

[2] Dargie W and Poellabauer C in (2010). Fundamentals of wireless sensor networks: Theory and practice. Wiley.

[3] Hung C, C Lin, K.C, J. Hsu, C Chou and Tu C in (2010). On enhancing network-lifetime using opportunistic routing in wireless sensor networks. 2010 Proceedings of 19th International Conference on Computer Communications and Networks (ICCCN).

[4] W.Xu, , K.Ma, W.Trappe and Y.Zhang in (2006). Jamming sensor networks: attack and defense strategies. IEEE Network.

[5] F.Cadger, K.Curran, J.Santos and S.Moffett in (2013). A survey of geographical routing in wireless ad-hoc networks. IEEE Communications Surveys & Tutorials.

[6] Di Tang, Tongtong Li, Jian Ren and Jie Wu in (2014). Cost-aware SEcure routing (CASER) protocol design for wireless sensor networks. IEEE Transactions on Parallel and Distributed Systems.

[7] Y.Li, J.Ren, and J.Wu in (2011). Quantitative measurement and design of source-location privacy schemes for wireless sensor networks. IEEE Transactions on Parallel and Distributed Systems.

[8] Sameh Zakhary, Milena Radenkovic and Abderrahim Benslimane in (2014). Efficient location privacy-aware forwarding in opportunistic mobile networks. IEEE Transactions on Vehicular Technology.

[9] Al-Sakib Khan Pathan, Hyung-Woo Lee and Choong Seon Hong in (2006). Security in wireless sensor networks: issues and challenges. Proceeding of the 8th International Conference on Advanced Communication Technology.

[10] Haibo Zhang and Hong Shen in (2009). Balancing energy consumption to maximize network lifetime in data-gathering sensor networks. IEEE Transactions on Parallel and Distributed. Systems.

[11] Y.Li and J.Ren in (2009). Preserving source-location privacy in wireless sensor networks. Proceedings of 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks.

[12] Y.Li and J.Ren in (2010). Source-location privacy through dynamic routing in wireless sensor networks. 2010 Proceedings IEEE INFOCOM.

[13] Y.Li, Y.Yang and X.Lu in (2010). Rules of designing routing metrics for greedy, face, and combined greedy-face routing. IEEE Transactions on Mobile Computing.

[14] F.Liu, C.Tsui and Y.J Zhang in (2010). Joint routing and sleep scheduling for lifetime maximization of wireless sensor networks. IEEE Transactions on Wireless Communications.

[15] Y.Li, J.Li, J.Ren and J.Wu in (2012). Providing hop-by-hop authentication and source privacy in wireless sensor networks. 2012 IEEE INFOCOM.

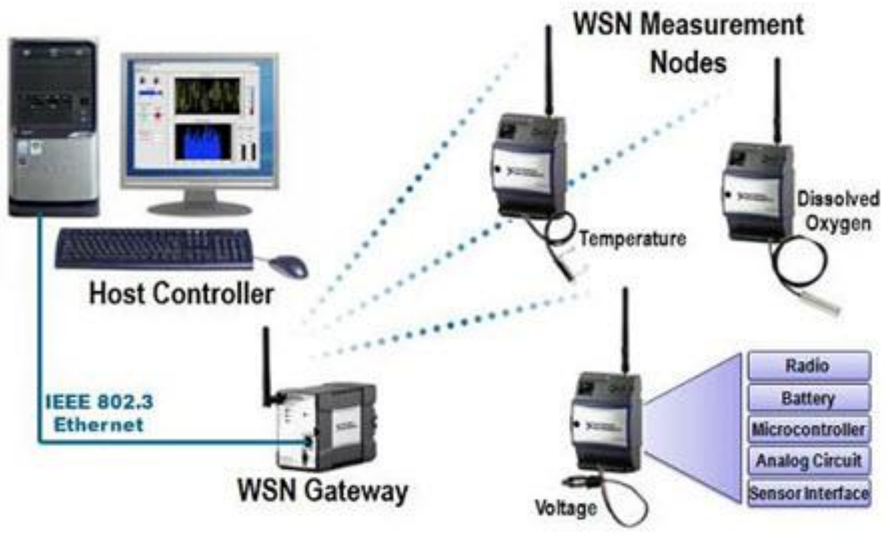
APPENDIX

WSN

A wireless sensor network (WSN) consists of three main components: nodes, gateways, and software. The spatially distributed measurement nodes interface with sensors to monitor assets or their environment. The acquired data is wirelessly transmitted to the gateway, which provides a connection to the wired world where you can collect, process, analyze, and present your measurement data using software. Routers are a special type of measurement node that you can use to extend distance and reliability in a WSN.

Basic WSN Architectures

For most WSN applications, you can create a basic network architecture in which the distributed measurement nodes acquire data from the world around them and transmit these measurements to a gateway, as seen in Figure 1.



This represents the WSN applications, you can create a network architecture in which the distributed measurement nodes gather data and transmit the measurements to a gateway.

Enhancing Your WSN Architecture

The NI platform helps you customize and enhance your WSN architecture even further. With the flexibility of Ethernet connectivity, you can add other devices and functionality to your WSN system. This can range from enterprise-level devices such as databases and servers to wired I/O, control systems, or third-party WSN products. LabVIEW Real-Time allows embedded data logging and open communication on the gateway, while the LabVIEW WSN Module allows node customization and local decision making at the node level.

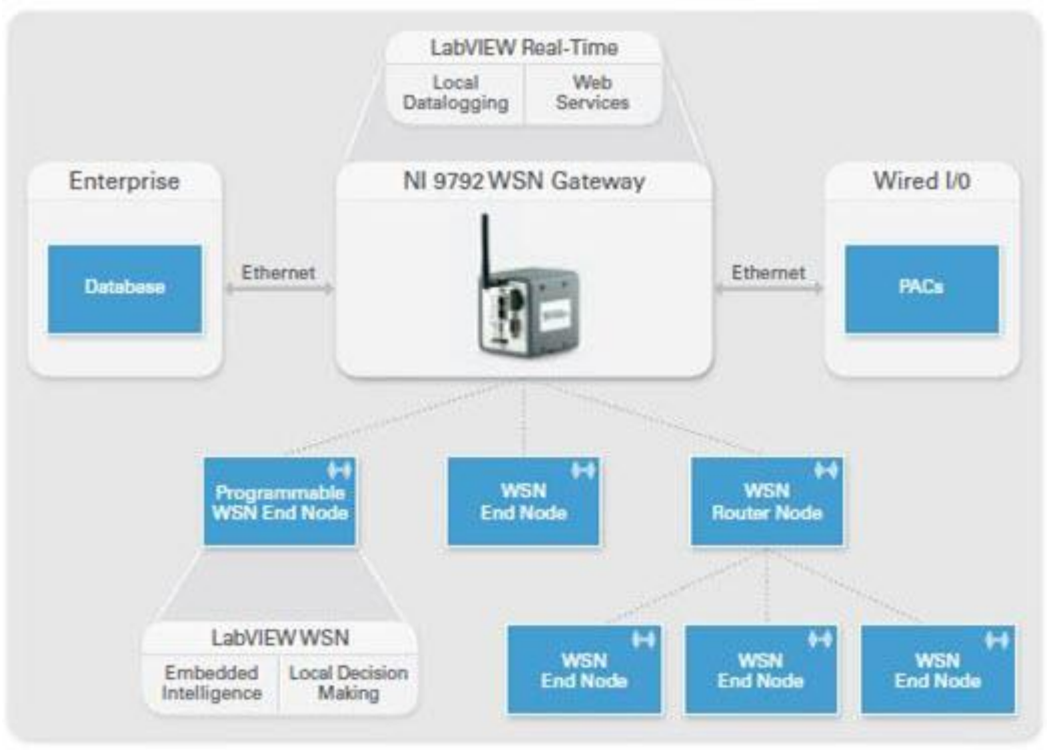


Figure 3. There are several hardware customization and software enhancement options for NI WSNs.

LabVIEW integration can enhance the capabilities of your wireless measurement system. Besides the advanced processing and visualization capabilities that are common in LabVIEW, LabVIEW Web services give you the opportunity to publish your measurement data to a Web server, such as the integrated Web server on the NI 9792, delivering convenient remote access to your WSN from virtually anywhere. With this complete system architecture, you can quickly and easily acquire data using an NI WSN, process and host that data on a server, and then access the data conveniently and remotely from a wireless smart device, such as an iPhone or laptop.

Measurement Nodes

NI WSN measurement nodes feature direct sensor connectivity, reliable communication, and industrial ratings. The devices are battery-powered, offering up to a three-year lifetime on four AA batteries, and you can combine them with NI outdoor enclosures for long-term outdoor deployment. You can also use LabVIEW drivers for WSN devices to add third-party WSN gateways and nodes to your measurement system. Each node is offered in a programmable and non-programmable version. With the programmable nodes, you can use the LabVIEW WSN Module and graphical programming to customize the node’s behavior, adding intelligence to perform local analysis and control

Node

A sensor node, also known as a mote (chiefly in North America), is a node in a sensor network that is capable of performing some processing, gathering sensory information and communicating with other connected nodes in the network. A mote is a node but a node is not always a mote.

Eclipse

Eclipse is an integrated development environment (IDE) used in computer programming, and is the most widely used Java IDE. It contains a base workspace and an extensible plug-in system for customizing the environment. Eclipse is written mostly in Java and its primary use is for developing Java applications, but it may also be used to develop applications in other programming languages via plugins, including Ada etc. It can also be used to develop documents with LaTeX (via a TeXlipse plug-in) and packages for the software Mathematics. Development environments include the Eclipse Java development tools (JDT) for Java and Scala, Eclipse CDT for C/C++, and Eclipse PDT for PHP, among others.

The initial codebase originated from IBM VisualAge. The Eclipse software development kit (SDK), which includes the Java development tools, is meant for Java developers. Users can extend its abilities by installing plug-ins written for the Eclipse Platform, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules. Since the introduction of the OSGi implementation (Equinox) in version 3 of Eclipse, plug-ins can be plugged-stopped dynamically and are termed (OSGI) bundles.

Java

Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture. As of 2016, Java is one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Gosling at Sun Microsystems(which has since been acquired by Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun relicensed most of its Java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java (bytecode compiler), GNU Classpath (standard libraries), and IcedTea-Web (browser plugin for applets).

Java jung

JUNG (the Java Universal Network/Graph Framework) was an open source graph modeling and visualization framework written in Java under the BSD license. The framework comes with a number of layout algorithms built in, as well as analysis algorithms such as graph clustering and metrics for node centrality.

JUNG's architecture is designed to support a variety of representations of entities and their relations, such as directed and undirected graphs, multi-modal graphs, graphs with parallel edges, and hypergraphs. It provides a mechanism for annotating graphs, entities, and relations with metadata. JUNG also facilitates the creation of analytic tools for complex data sets that can examine the relations between entities as well as the metadata attached to each entity and relation. JUNG includes implementations of a number of algorithms from graph theory, data mining, and social network analysis, such as routines for clustering, decomposition, optimization, random graph generation, statistical analysis, and calculation of network distances, flows, and importance measures.

JUNG provides a visualization framework that makes it easy to construct tools for the interactive exploration of network data. Users can use one of the layout algorithms provided, or use the framework to create their own custom layouts. In addition, filtering mechanisms are provided which allow users to focus their attention, or their algorithms, on specific portions of the graph.

Jframe

JFrame is Model View Controller Model2 framework for Web based applications written in Java language. It is very simple and emphasis on security and stability. Because of it simplicity this framework can be used and for studying purposes.

Swing is a GUI widget toolkit for Java. It is part of Oracle's Java Foundation Classes (JFC) – an API for providing a graphical user interface (GUI) for Java programs.

Swing was developed to provide a more sophisticated set of GUI components than the earlier Abstract Window Toolkit (AWT). Swing provides a look and feel that emulates the look and feel of several platforms, and also supports a pluggable look and feel that allows applications to have a look and feel unrelated to the underlying platform. It has more powerful and flexible components than AWT. In addition to familiar components such as buttons, check boxes and labels, Swing provides several advanced components such as tabbed panel, scroll panes, trees, tables, and lists.

Unlike AWT components, Swing components are not implemented by platform-specific code. Instead, they are written entirely in Java and therefore are platform-independent. The term "lightweight" is used to describe such an element.

Dijkstra's algorithm

Dijkstra's algorithm is an algorithm for finding the shortest paths between nodes in a graph, which may represent, for example, road networks. It was conceived by computer scientist Edsger W. Dijkstra in 1956 and published three years later.

The algorithm exists in many variants; Dijkstra's original variant found the shortest path between two nodes, but a more common variant fixes a single node as the "source" node and finds shortest paths from the source to all other nodes in the graph, producing a shortest-path tree.

For a given source node in the graph, the algorithm finds the shortest path between that node and every other. It can also be used for finding the shortest paths from a single node to a single destination node by stopping the algorithm once the shortest path to the destination node has been determined. For example, if the nodes of the graph represent cities and edge path costs represent driving distances between pairs of cities connected by a direct road, Dijkstra's algorithm can be used to find the shortest route between one city and all other cities. As a result, the shortest path algorithm is widely used in network routing protocols, most notably IS-IS (Intermediate System to Intermediate System) and Open Shortest Path First (OSPF). It is also employed as a subroutine in other algorithms such as Johnson's.

Let the node at which we are starting be called the initial node. Let the distance of node Y be the distance from the initial node to Y. Dijkstra's algorithm will assign some initial distance values and will try to improve them step by step.

1. Mark all nodes unvisited. Create a set of all the unvisited nodes called the unvisited set.
2. Assign to every node a tentative distance value: set it to zero for our initial node and to infinity for all other nodes. Set the initial node as current.
3. For the current node, consider all of its unvisited neighbors and calculate their tentative distances through the current node. Compare the newly calculated tentative distance to the current assigned value and assign the smaller one. For example, if the current node A is marked with a distance of 6, and the edge connecting it with a neighbor B has length 2, then the distance to B through A will be 6 + 2 = 8. If B was previously marked with a distance greater than 8 then change it to 8. Otherwise, keep the current value.
4. When we are done considering all of the neighbors of the current node, mark the current node as visited and remove it from the unvisited set. A visited node will never be checked again.
5. Move to the next unvisited node with the smallest tentative distances and repeat the above steps which check neighbors and mark visited.

6. If the destination node has been marked visited (when planning a route between two specific nodes) or if the smallest tentative distance among the nodes in the unvisited set is infinity (when planning a complete traversal; occurs when there is no connection between the initial node and remaining unvisited nodes), then stop. The algorithm has finished.
7. Otherwise, select the unvisited node that is marked with the smallest tentative distance, set it as the new "current node", and go back to step 3.

When planning a route, it is actually not necessary to wait until the destination node is "visited" as above: the algorithm can stop once the destination node has the smallest tentative distance among all "unvisited" nodes (and thus could be selected as the next "current").

Cluster head

Clustering is one of the important methods for prolonging the network lifetime in wireless sensor networks (WSNs). It involves grouping of sensor nodes into clusters and electing cluster heads (CHs) for all the clusters. CHs collect the data from respective cluster’s nodes and forward the aggregated data to base station. A major challenge in WSNs is to select appropriate cluster heads. In this paper, we present a fuzzy decision-making approach for the selection of cluster heads. Fuzzy multiple attribute decision-making (MADM) approach is used to select CHs using three criteria including residual energy, number of neighbors, and the distance from the base station of the nodes. The simulation results demonstrate that this approach is more effective in prolonging the network lifetime than the distributed hierarchical agglomerative clustering (DHAC) protocol in homogeneous environments.

Grid

In graphic design, a grid is a structure (usually two-dimensional) made up of a series of intersecting straight (vertical,horizontal, and angular) or curved guide lines used to structure content. The grid serves as an armature or framework on which a designer can organize graphic elements (images, glyphs, paragraphs, etc.) in a rational, easy-to-absorb manner. A grid can be used to organize graphic elements in relation to a page, in relation to other graphic elements on the page, or relation to other parts of the same graphic element or shape.

Routing table

In computer networking a routing table, or routing information base (RIB), is a data table stored in a router or a networked computer that lists the routes to particular network destinations, and in some cases, metrics (distances) associated with those routes. The routing table contains information about the topology of the network immediately around it. The construction of routing tables is the primary goal of routing protocols. Static routes are entries made in a routing table by non-automatic means and which are fixed rather than being the result of some network topology "discovery" procedure.

A routing table uses the same idea that one does when using a map in package delivery. Whenever a node needs to send data to another node on a network, it must first know where to send it. If the node cannot directly connect to the destination node, it has to send it via other nodes along a proper route to the destination node. Most nodes do not try to figure out which route(s) might work; instead, a node will send an IP packet to a gateway in the LAN, which then decides how to route the "package" of data to the correct destination. Each gateway will need to keep track of which way to deliver various packages of data, and for this it uses a Routing Table. A routing table is a database which keeps track of paths, like a map, and allows the gateway to provide this information to the node requesting the information.

With hop-by-hop routing, each routing table lists, for all reachable destinations, the address of the next device along the path to that destination: the next hop. Assuming that the routing tables are consistent, the simple algorithm of relaying packets to their destination's next hop thus suffices to deliver data anywhere in a network. Hop-by-hop is the fundamental characteristic of the IP Internetwork Layer and the OSI Network Layer.

The primary function of a router is to forward a packet toward its destination network, which is the destination IP address of the packet. To do this, a router needs to search the routing information stored in its routing table.

A routing table is a data file in RAM that is used to store route information about directly connected and remote networks. The routing table contains network/next hop associations. These associations tell a router that a particular destination can be optimally reached by sending the packet to a specific router that represents the "next hop" on the way to the final destination. The next hop association can also be the outgoing or exit interface to the final destination.

The network/exit-interface association can also represent the destination IP address of the IP packet. This association occurs on the router's directly connected networks.

A directly connected network is a network that is directly attached to one of the router interfaces. When a router interface is configured with an IP address and subnet mask, the interface becomes a host on that attached network. The network address and subnet mask of the interface, along with the interface type and number, are entered into the routing table as a directly connected network. When a router forwards a packet to a host, such as a web server, that host is on the same network as a router's directly connected network.

A remote network is a network that is not directly connected to the router. In other words, a remote network is a network that can only be reached by sending the packet to another router. Remote networks are added to the routing table using either a dynamic routing protocol or by configuring static routes. Dynamic routes are routes to remote networks that were learned automatically by the router, using a dynamic routing protocol. Static routes are routes to networks that a network administrator manually configured.