SENSOR NETWORKS TO MONITOR QUALITY OF RURAL LIFE

Project report submitted in fulfillment of the requirement for the degree of Bachelor of Technology

In

Computer Science and Engineering

By

Anshul Saxena (121305)
Rajeev Agarwal (121312)

Under the supervision of

Dr. Yashwant Singh

to

Department of Computer Science & Engineering and Information Technology
Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

CANDIDATE'S DECLARATION

I hereby declare that the work presented in this report entitled **"Sensor Networks To Monitor Quality Of Rural Life"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology, Waknaghat is an authentic record of my own work carried out over a period from August 2015 to December 2015 under the supervision of **Dr. Yashwant Singh,** Assc. Professor, Computer Science And Engineering. The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Anshul Saxena	Rajeev Agarwal
(121305)	(121312)

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Dr. Yashwant Singh
Associate Professor
Computer Science And Engineering
Dated:10/06/2016

ACKNOWLEGEMENT

We are grateful and indebted to Dr. Yashwant Singh , Associate Professor, Department of Computer Science And Engineering for his help and advice in completion of this project report. We also express our deep sense of gratitude and appreciation to our guide for his constant supervision, inspiration and encouragement right from the beginning of this Seminar report. We deem it a pleasant duty to place on record our sincere and heartfelt gratitude to our project guide for his long sightedness, wisdom and co-operation which helped us in tackling crucial aspects of the project in a very logical and practical way.

Anshul Saxena	Rajeev Agarwal
(121305)	(121312)
Computer Science & Engineering	Computer Science & Engineering

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The rural area in India has 68.84% population as compared to the 31.16% urban population. Agriculture is the main source of income in rural areas of India. Animals like cow, buffalo, sheep, goat etc play a significant role in lives of rural India. They are used as a source of income. Hence animal husbandry becomes a major concern.

This report discusses the issues and concerns related to the health of animals with respect to the rural areas of Himachal Pradesh based on the survey done by us. After discovering the concern we have proposed possible solution for the said issues. The solution uses the upcoming technology of Wireless Sensor Network (WSN). It shows how the upcoming technology can be integrated seamlessly in the rural environment to improve and uplift the life style of people staying in rural area of Himachal Pradesh.

Then this report tries to analyse the trend of health issues in livestock and then based on that it tries and give a prediction of health issues that the cattle may face in near future. It takes into consideration various indices like THI to look after the health of cattle. Also a comparative study was done to decide the language for coding the given problem statement and details of which the following report contains.

CONTENTS

CHAPTER- 1
INTRODUCTION

1.1 INTRODUCTION

The phenomenal advancement in technologies like micro-electromechanical systems (MEMS), very large scale integration (VLSI) and wireless communications which facilitates the growth of low cost, low power, multifunctional sensor nodes, which are small in size with limited processing and computing resources and they are inexpensive[1]. These features have made WSNs popular and these are being used in many real life problems.

1.1.1 Wireless Sensor Networks

A wireless sensor network is a group of specialized transducers with a communications infrastructure for monitoring and recording conditions at diverse locations. Commonly monitored parameters are temperature, humidity, pressure, wind direction and speed, illumination intensity, vibration intensity, sound intensity, power-line voltage, chemical concentrations, pollutant levels and vital body functions.

A sensor network consists of multiple detection stations called sensor nodes, each of which is small, lightweight and portable. Every sensor node is equipped with a transducer, microcomputer, transceiver and power source. The transducer generates electrical signals based on sensed physical effects and phenomena. The microcomputer processes and stores the sensor output. The transceiver receives commands from a central computer and transmits data to that computer. The power for each sensor node is derived from a battery.

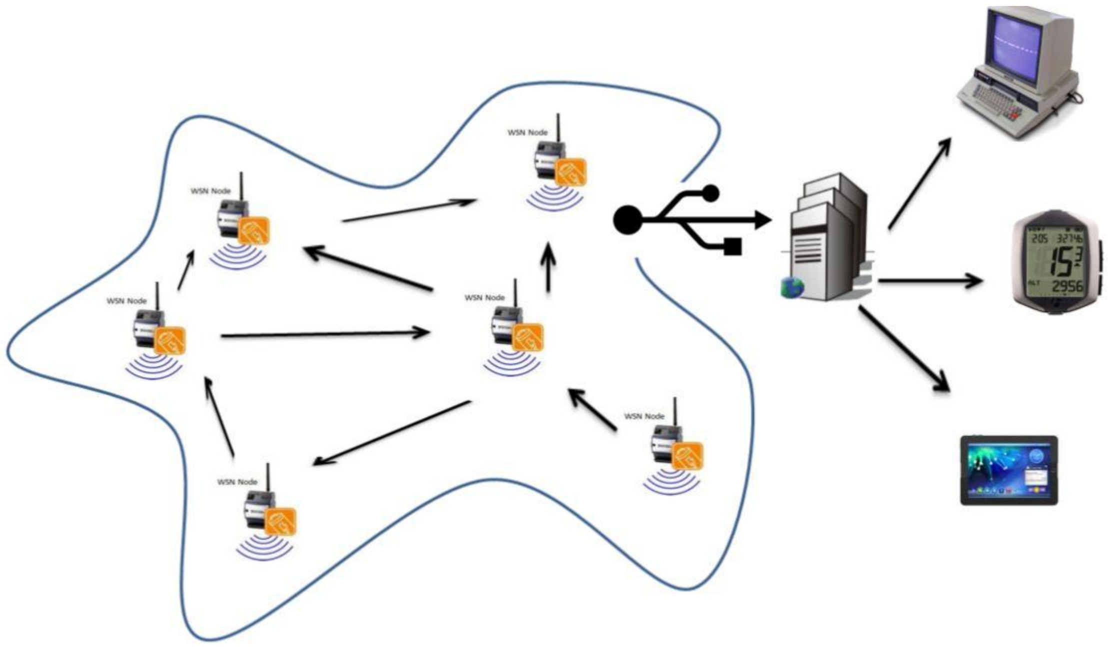


Figure 1: Wireless Sensor Network [1]

1.1.2 Applications of WSNs

Engineers have created WSN applications for areas including health care, utilities, and remote monitoring. In health care, wireless devices make less invasive patient monitoring and health care possible. For utilities such as the electricity grid, streetlights, and water municipals, wireless sensors offer a lower-cost method for collecting system health data to reduce energy usage and better manage resources. Remote monitoring covers a wide range of applications where wireless systems can complement wired systems by reducing wiring costs and allowing new types of measurement applications. Remote monitoring applications include:

- Environmental monitoring of air, water, and soil
- Structural monitoring for buildings and bridges
- Industrial machine monitoring
- Process monitoring
- Asset tracking

1.1.3 Methodology

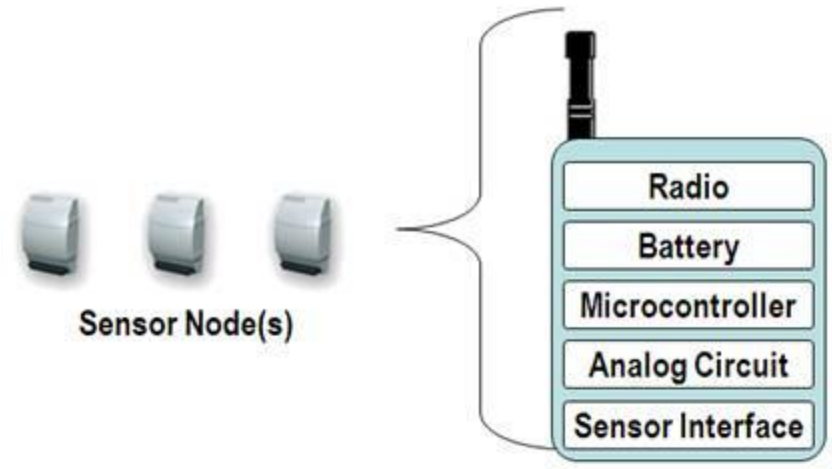
- Read and analyze existing literature.
- Compare various algorithms on the basis of cost, accuracy, efficiency.
- Implement and test the most efficient algorithm using JAVA.
- Modify the algorithm.
- Implement the modified algorithm.

1.1.4 Components of a WSN Node

A WSN node contains several technical components. These include the radio, battery, microcontroller, analog circuit, and sensor interface. When using WSN radio technology, you must make important trade-offs. In battery-powered systems, higher radio data rates and more frequent radio use consume more power. Often three years of battery life is a requirement, so many of the WSN systems today are based on ZigBee due to its low-power consumption. Because battery life and power management technology are constantly evolving and because of the available IEEE 802.11 bandwidth, Wi-Fi is an interesting technology.

The second technology consideration for WSN systems is the battery. In addition to long life requirements, you must consider the size and weight of batteries as well as international standards for shipping batteries and battery availability. The low cost and wide availability of carbon zinc and alkaline batteries make them a common choice.

To extend battery life, a WSN node periodically wakes up and transmits data by powering on the radio and then powering it back off to conserve energy. WSN radio technology must efficiently transmit a signal and allow the system to go back to sleep with minimal power use. This means the processor involved must also be able to wake, power up, and return to sleep mode efficiently. Microprocessor trends for WSNs include reducing power consumption while maintaining or increasing processor speed. Much like your radio choice, the power consumption and processing speed trade-off is a key concern when selecting a processor for WSNs. This makes the x86 architecture a difficult option for battery-powered devices.



1.2 PROBLEM STATEMENT

- ➨ **Wireless Sensor Networks to monitor quality of rural life.**
- Monitoring health of animals and humans with respect to the rural areas.
- Instrumenting natural spaces can enable us to do long term data collection.
- Data collected needs to be processed and refined.
- Various algorithms to cater to the data.
- Cost, complexity, efficiency and reliability always remains a question.

1.3 OBJECTIVES

Following are the objectives of our project-

- To undergo extensive survey on existing literature.
- To analyze the various existing problems with rural life in Himachal Pradesh.
- To provide efficient solution to the given problem.
- To recognize different sensors for various problems.
- To implement various algorithms.

1.4 VARIOUS RURAL LIFE PROBELMS

Rural India is the backbone of Indian economy and the daily needs of whole of India is met by these rural areas. Still when it comes to technological advancement in this area the outcome is almost null. There is a need to address the health issues of cattle and rural population with the help of latest technology and to that we are using WSN.

Rural areas suffers from many problems and some of them include-

- Human and Animal Healthcare during floods.
- Animal Health Issues in Rural Area.
- Human and Animal Healthcare during Landslides.
- Locals Health Issues in Rural Area.

1.5 ORGANISATION

This paper is organized as follows: In Chapter 2, we have given an overview of the research papers we have read. The Chapter 3 briefs about the various system development phases of the project. The Chapter 4 contains the analysis of the proposed algorithm. And we conclude our discussion in Chapter 5, further discussing about the future scope of the project.

CHAPTER-2
LITERATURE SURVEY

To get familiar with the given topic, we studied various research papers that have been already proposed. We collected various papers related to the topic to gather all the information about the wireless sensor networks and the different attacks associated with it. Following is the abstract of few papers that we surveyed:

2.1 Maneesha V. Ramesh, Sangeeth Kumar, and P. Venkat Rangan : "WSN for Landslide Detection"

Real-time monitoring of environmental disasters are one of the prime necessity of the world. Different technologies have been developed for this purpose. Wireless sensor networks (WSN) are one of the major technologies that can be used for real-time monitoring. WSN has the capability of large scale deployment, low maintenance, scalability, adaptability for different scenarios etc. WSN has its own limitation such as low memory, power, bandwidth etc., but its capability to be deployed in hostile environment, and low maintenance requirement made it one of the best suited technologies for real-time monitoring.[1] This paper discusses the design and deployment of a landslide detection system using wireless sensor network.

LANDSLIDES

Once a landslide is triggered, material is transported by various mechanisms including sliding, flowing and falling.

The types of landslides vary with respect to the:

- Rate of movement: This ranges from a very slow creep (millimeters/year) to extremely rapid (meters/second).

Type of material:

– Landslides are composed of bedrock, unconsolidated sediment and/or organic debris.

_ Nature of movement:

– The moving debris can slide, slump, flow or fall.

Landslides constitute a major natural hazard in India that accounts for considerable loss of life and damage to communication routes, human settlements, agricultural fields and forest lands. The Indian subcontinent, with diverse physiographic, seismotectonic and climatological conditions is subjected to varying degree of landslide hazards; the Himalayas including Northeastern mountains ranges being the worst affected, followed by a section of Western Ghats and the Vindhyas.[1]

Sensors Needed for Monitoring Rainfall Induced Landslides

Under heavy rainfall conditions, rain infiltration on the slope causes instability, a reduction in the factor of safety, transient pore pressure responses, changes in water table height, a reduction in shear strength which holds the soil or rock, an increase in soil weight and a reduction in the angle of repose. When the rainfall intensity is larger than the slope saturated hydraulic conductivity, runoff occurs

.

Three distinct physical events occur during a landslide:
_ the initial slope failure,
_ the subsequent transport, and
_ the final deposition of the slide materials.

The initial slope failure can occur due to the increase in pore pressure and soil moisture content, under heavy rainfall, which necessitates the inclusion of geophysical sensors for detecting the change in pore pressure and moisture content with the warning system developed for landslide detection. So the system discussed in this paper also includes geophysical sensors such as pore pressure transducer and dielectric moisture sensor for capturing the in-situ measurements. After the slope failure the subsequent transport of the material happens that will generate slope gradient change, vibration etc., which has to be measured and monitored for effective issue of warning. So the warning system includes strain gauge, and tiltmeter, that can be used for measuring in-situ slope gradient changes. Along with them geophone is used for analyzing the vibration.

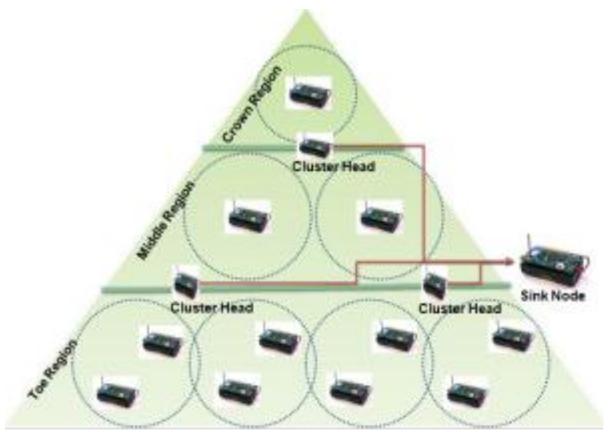


Figure 1: Regionalized Wireless Sensor Network Architecture for Landslides[1]

Wireless Sensor Network Algorithm

The wireless sensor network uses four algorithms for implementing clustering, distributed consensus among the data, energy efficient data aggregation and time synchronization,

which will contribute for the development of an efficient landslide detection system. The real-time monitoring networks are constrained by energy consumption, due to the remote location of the deployment site and the non-availability of constant power. Considering these factors, the wireless sensor network at the deployment site implements a totally innovative concept for distributed detection, estimation and consensus to arrive at reliable decisions, more accurate than that of each single sensor and capable to achieve globally optimal decisions.

The different methods that can be used for implementing this **algorithm**, for landslide scenario are:

1) Homogeneous sensor columns deployed in each region can be compared and a consensus value can be achieved for all the sensor columns in that region

2) All the sensors deployed in the landslide prone area can be assigned with a weightage with regard to its impact on landslide detection, and a common consensus value can be achieved executing the algorithm at once, for all deployed sensors

3) Decentralized consensus performed for the same type of sensors in all sensor columns in a region

Decentralized consensus for the same type of sensors has been developed for the deployed network. The decentralized algorithm will be executed for each type of sensors, one by one, for all homogeneous sensor columns deployed at each region. After initial set of sensors achieve its consensus, the next set of sensors will execute the decentralized algorithm and so on, as shown in Figure 3. The other designs demand priory knowledge of correlation between different geophysical sensors, whereas this method does not require this priory knowledge, but the processing delay will be more compared to other methods, due to the multiple execution of same algorithm.

CHALLENGES

Wireless sensor network for landslide detection is one of the challenging research areas available today in the field of geophysical research. This paper describes about an actual field deployment of a wireless sensor network for landslide detection. This system uses a heterogeneous network composed of wireless sensor nodes, Wi-Fi, and satellite terminals for efficient delivery of real time data to the data management center. The data management center is equipped with softwares and hardwares needed for sophisticated analysis of the data. The results of the analysis in the form of landslide warnings and risk assessments will be provided to the inhabitants of the region. The pilot deployment of this system is already in place at Anthoniar Colony, Munnar, Idukki, Kerala, India. In the future, this work will be extended to a full deployment with increased spatial variabilty, and the work in this regard is progressing. Field experiments will be conducted to determine the effects of density of the nodes, vegetation, location of sensor columns etc., for detecting rainfall induced landslides, that may help in the development of low cost wireless sensor network for landslide detection.[1]

2.2 Ankit R. Bhavsar and Harshal A. Arolkar, **Wireless Sensor Networks : A Possible Solution for Animal Health Issues in Rural Areas Of Gujarat: A Journal, Vol.2 Issue 2 July 2012, 14 pages**

The paper discusses the issues and concerns related to the health of animals with respect to the rural areas of Gujarat based on the survey done by them. After discovering the concern they have proposed possible solution for the said issues. The solution uses the upcoming technology of Wireless Sensor Network (WSN). It shows how the upcoming technology can be integrated seamlessly in the rural environment to improve and uplift the lifestyle of people staying in rural area of Gujarat.

A Wireless Sensor Network (WSN) is an infrastructure comprised of sensing, computing and communication elements that gives us the ability to observe and react to events in a specified environment. Basically Wireless Sensor Network (WSN) has four components: a assembly of distributed sensors, an interconnecting wireless network, a central point of information gathering, and a set of computing resources at the central point. It gives administrator ability to instrument, observe and react to events in a specified environment. Usage and deployment of wireless sensor technology has increased in several areas within industry as well as day to day life. WSN are used in application like health care, monitoring, tracking and sensing.
According to the last Livestock Census done in 2007, the total livestock population (excluding dogs and stray animals) in the state was 235.25 lakhs, registering an increase of 8.55 percent over the year 2003. As per the census there were 79.76 lakhs cattle, 87.74 lakhs buffalo, 20.02 lakhs sheep, 46.40 lakhs goats and 1.34 lakhs other animals. The percentages graph of total population is shown in Figure 1.

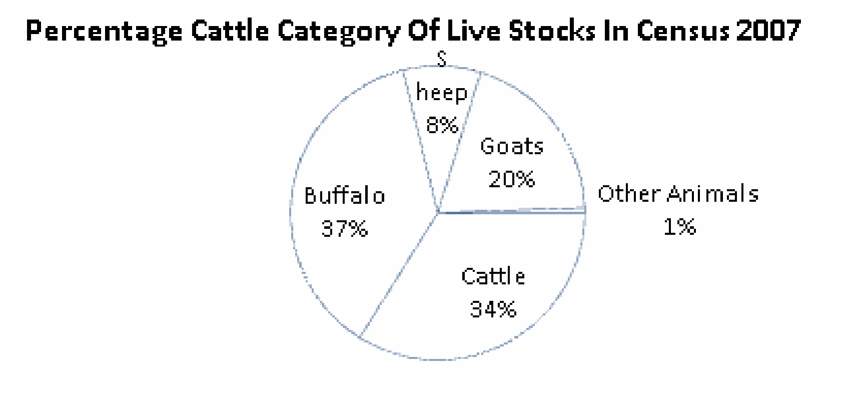


Figure 1: Percentage cattle category of live stocks after 2007 census[2]

To find the actual scenario of the problem related to animals and animal health issues, they did a survey in rural areas of the state of Gujarat. The Survey was spread across 3 districts, 6 talukas and 19 villages, A total of forty questions related to animals and there well being were framed for the survey, The partial outcome of the survey is given in the section. The results mentioned are mainly related to milk producing capacity of health of the animal. After conducting the survey they found that 98% of the surveyors agree to the fact that milk production decreases when the animal falls ill. Thus it was observed that majority of the villagers believe that illness of the animal also affects its milk producing capacity, As the milk production was getting affected due to illness, they tried to find out the season during which the livestock falls ill majority of the time, Figure 2 shows the outcome of this question.[2]

The next thing they needed to identify was the remedial action that the owners took in case of illness of livestock. Figure 3 shows that out of 140 persons surveyed 124 took advice of doctor.

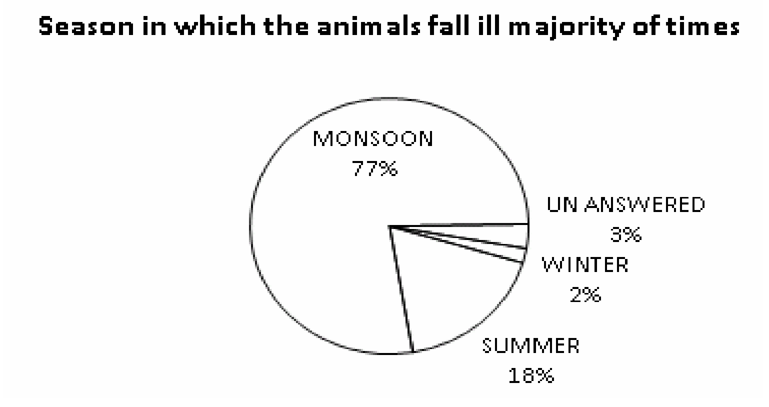


Figure 2: Season when animals fall ill majority of time[2]

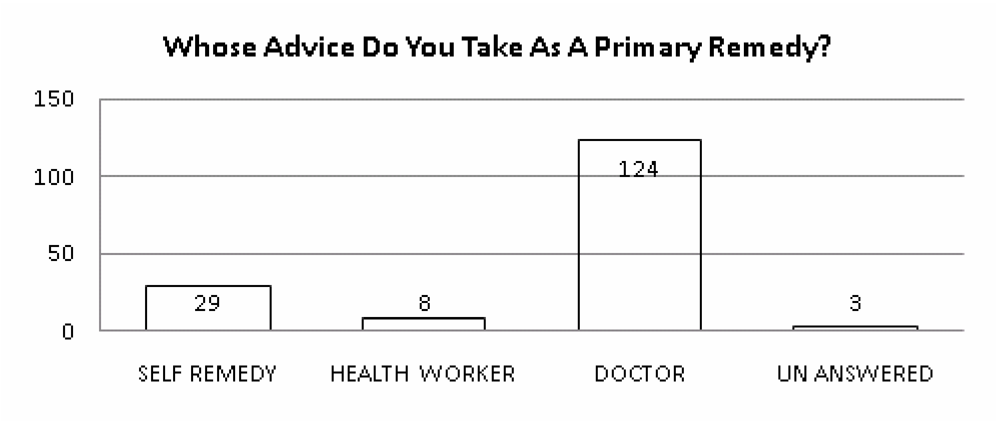


Figure 3: Whose advice is preferred in case of illness[2]

As the people prefer to take advice of doctors it is necessary to take the animal to the doctor or a doctor's should visit the animal. In both cases the distance between hospital/dispensary and the ill animals is of significance[2]. Figure 4 show that 31% of the people surveyed take the animal to the doctor by walk. This may result in deterioration of health condition of the animal.

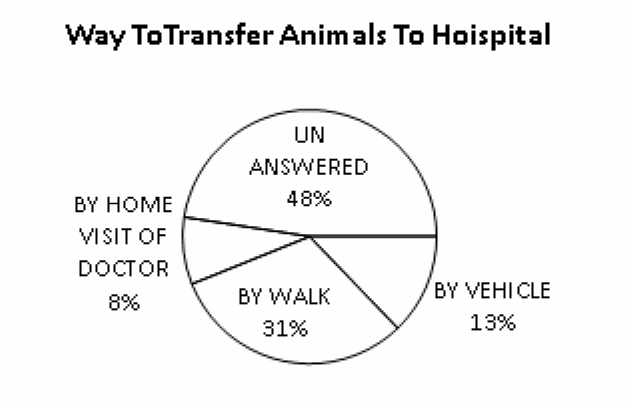


Figure 4: Way of Transfer Animal to Hospital[2]

To implement their proposed solution they needed to know the comfort level of villagers in use of technology. It was found that 77% villagers (animal owners) have mobile phone and use it for voice call as well as sending and receiving SMS. Table 1 summarizes the outcome of the survey.

Issue	Answer	Percentage
Average no of animal with each family	1 to 5	72%
Milk production is reduced during illness of animal	Yes	92%
Season in which animal falls ill time	Monsoon	77%
Primary advice taken from whom at time of illness	Doctor	75%
Veterinary setup (hospital) in village	No	63%
Way of transportation of animal at the time of illness	By Walking	31%

Table 1: Summarizes the outcome of the survey[2]

The monitoring has to be done on continuous basis and will not interfere with the movement of the animals. It will also make it convenient for the owner as well as health worker to keep track of physical as well as environmental concerns[2]. In case of any eventualities like increase in body temperature, heart beats etc, a message informing the change can be generated instantly and send to either the owner, health worker or doctor and thus timely action can be taken to improve the health of animal.

The system consists of heterogeneous wireless sensor devices capable of sensing and transferring data. The devices when combined will form a network.[2] This network would be capable of collecting, aggregating, processing the data collected on occurrence of various events. Once the data is available, proper analysis of the data can be done and the healthcare provider (Veterinary Doctor) can be informed about the health status of the animals if required. The proposed architecture is as shown in Figure 5.

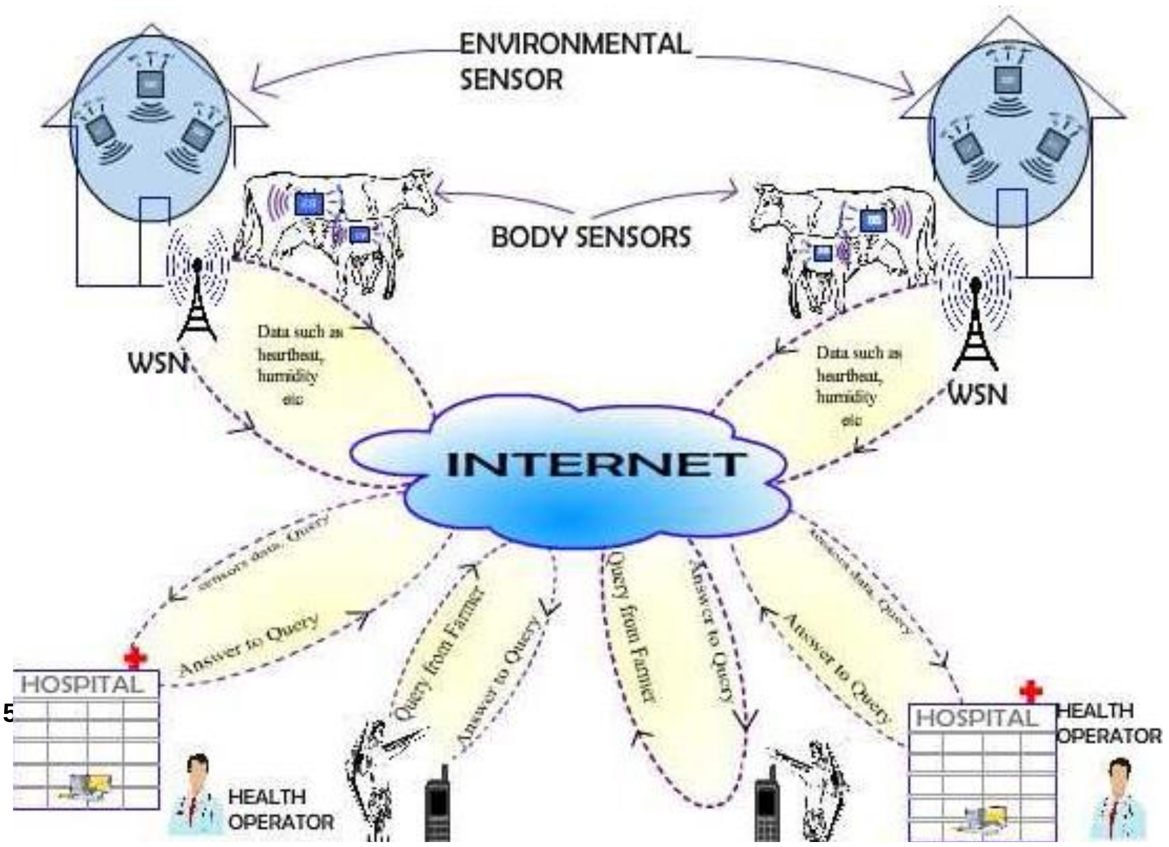


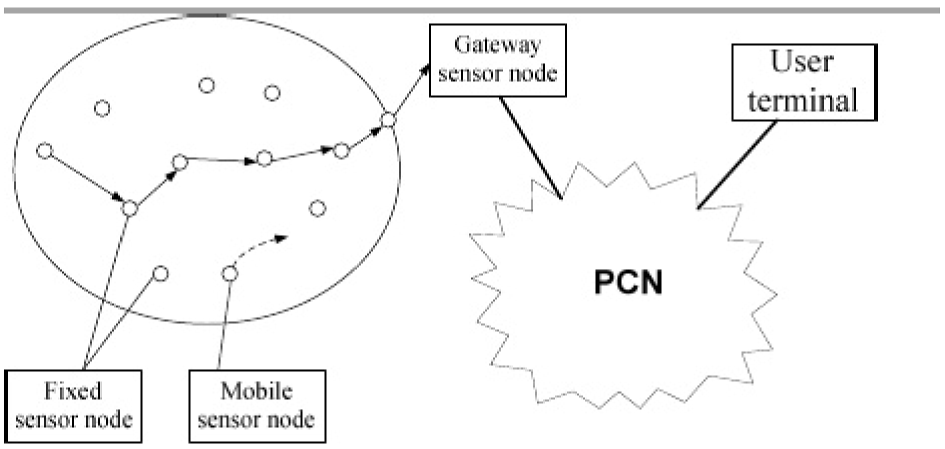
Figure 5

Figure 5: Proposed Architecture[2]

***2.3 Rashid Hussain, Ankit R. Bhavsar and Harshal A. Arolkar,* Use of Wireless Sensor Network for Human and Animal Health Care during Flood, Vol.2 Issue 3, March 2013, 5 pages**

Wireless sensor network is basically a collection of a large number of self-powered small sensing nodes that gather information from the field where they are deployed and communicate that information in a wireless fashion and finally handing over their processed data to a base station.
Wireless Sensor Networks are used because of the numerous advantages like low cost, wide coverage of the experimental area, real time data access and self-configuration.

A lot of research work has been going on in the recent times on the wireless sensors and health department. A WSN healthcare system consists of three main parts: body and home environment sensor network, access devices like gateway and public communication networks and care takers like remote central server, doctors, nurses and relatives. The gateway is used to bridge the gap between the WSN and the public communication networks. That is they are used to work as the impedance matching devices, protocol translators and rate convertors between them. In the health care system the sensors are attached to the body of the patient under observation. All the data collected from his body is wirelessly transferred to the head node and from there it is sent to the central processor for further processing.



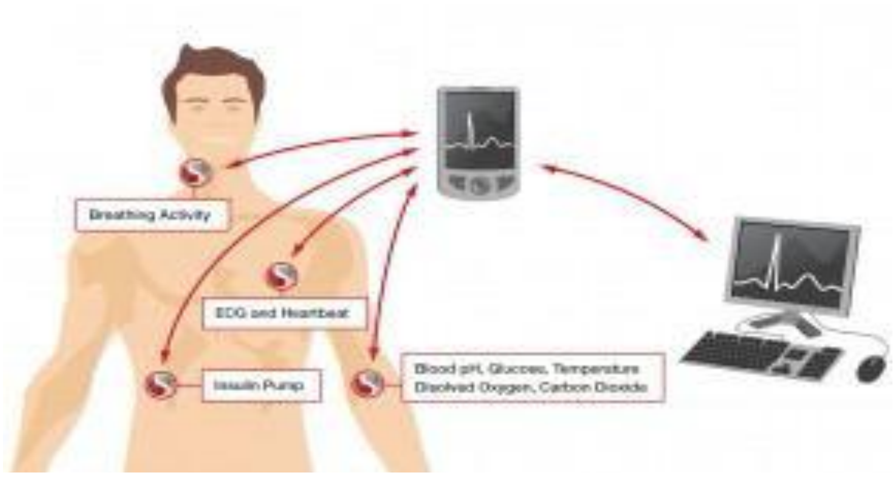

Breathing Activity
ECG and Heartbeat
Insulin Pump
Blood pH, Glucose, Temperature
Dissolved Oxygen, Carbon Dioxide

Figure 1:Body Sensors

The figure above show the different body sensors that are used for the observations. These sensors do not harm the patient at all and are thus human friendly. The different sensors from the body directly provide the required signals. These sensors are wearable and they also enable mobile health which is also known as m-health.[3] Thus an individual can use these sensors even while working and doing his daily activities. The simplest kind of physiological sensors includes Electrocardiograms (ECG), Electromyogram (EMG) and the motion sensors.

All the data that is taken from the sensors is in the raw form. This data is sent to the processors usually the computers at a hospital or the nursing homes for observation. After this the processed data is sent to the doctors or the relatives/care takers through a gateway. And this way a proper action can be taken within time. A personal server application can be run on the mobile phones also so that a care taker/relative can get the results directly from the processed data. The communication between personal server and the internet gateway take place using one of the standard data networks like cellular and /or WLAN.

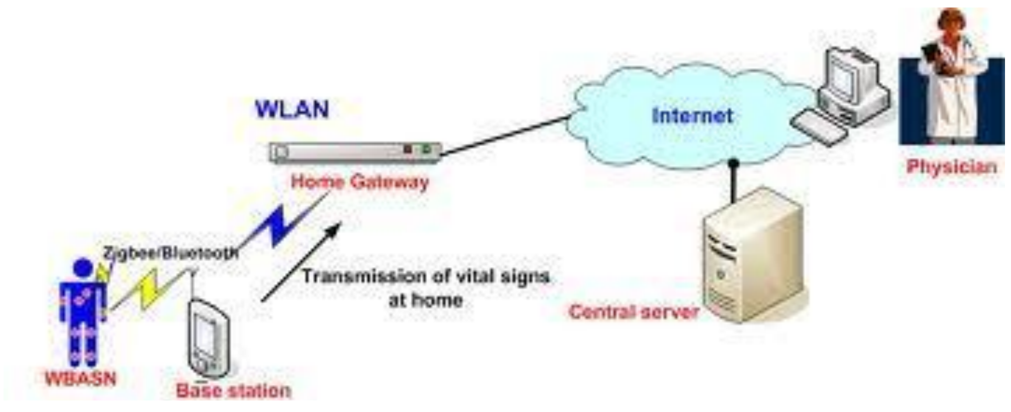


Figure 2:Flow of data

The Technology for smart sensors is still in its basic level yet its use can be seen at increasing pace in some of the fields like ECG monitoring, stress monitoring, asthma, emergency response and the post operation period.[3]

For the continuous monitoring of the health the body sensors are deployed on the human body. They are the miniaturized wearable sensors for continuous monitoring. Also some researchers are working on such kind of body sensors that are able to recognize the body on which they are deployed.

Effect of flood water on Human Health:

Consumption of the bacterial water can cause a range of conditions including diarrheal diseases, skin diseases, soft tissue infection, fever, or sometimes long term diseases also might be caused. Cuts or scratches acquired in the flood affected area can also be harmful. Flood effected river water is consumed by the human being living in that area. And often they don’t have knowledge about the diseases or bacteria present in the river water.[2] The rural areas do not have good medical facilities also so consumption of this river water often results in the death of people.

Effect of flood water on Animal Health:

Animal play an important role in the rural areas. Farmers depend heavily on them for the agricultural processes, for milk or meat. If the animal itself consumes bacteria effected flood water in heavy amounts then this can lead to the health problems of the animal[2]. If the milk or meat of this animal is consumed then it creates health problems for the human also.

Effect of flood water on Agriculture:

The flood water contains many kind of salts. These salts may be harmful for the agricultural lands and the crops growing in them. Once it is known that the water contains too much of salt then the water supply can be completed by other sources till the water of the river is fit for use.

The Water sensors are used in the water body that is to be taken under observation. Here we make use of two kinds of sensors for the detection of the water that is completely unfit for human as well as animal consumption.

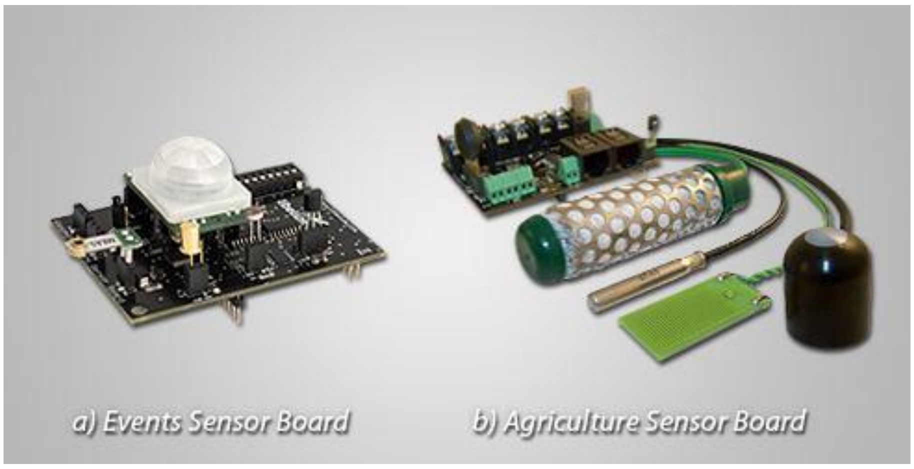
Wireless Sensor Network are a cost effective and scalable method for detecting early flood signs, forecasting floods and also monitoring the flood areas. Here they discuss the after situation of the river which renders the water unfit for use. The motes of the network are placed along the course of the river. These sensors can also monitor the weather conditions and rainfall making the forecasting of the floods easier.

Motes that are placed along the course of the river measure the increased water levels and generate alerts wirelessly by SMS or the Internet Database position. Waspmote measures these parameters using events sensor board (water level) and the agriculture sensor board (weather condition). Waspmote gets outstanding radio links in the frequency bands of 2.4 GHz, 900MHz and 868Mhz using Zigbee protocols.[3]

Thus using these sensors a large area can be surveyed. Till the water level is still high the river water contains the flood water in it.[3] Waspmote detects it and conveys the data to the base controller. Thus we come to know that the river still contains the flood water and is unfit for use.

We use another kind of sensors that directly detects the presence of algae or any other such substance in small amounts of water. As we come to know that the water of the river contains the flood water an alarm goes off. After this the water from the river is collected and then subjected to another sensor EOS-300. This sensor detects the presence of any kind of unfit substance in the water. So this way we can detect the presence of the unfit substances in water which can be consumed by the people living in the rural areas near the river. Thus the water will be declared unfit for drinking directly. And it will be consumable only after its purification. So the diseases that are caused because of drinking this water can be avoided.

Attached to the river there will be a small area that will be connected to the river itself. Now as the flood water enters the river, the level in that small portion will also increase, Sensors will sense this increase in water level. Through a pipe, this small portion containing the water will be connected to the testing systems. As the water level rises, the sensors will sense this and open the gates of the pipe through which the water will go straight away to the testing plant. The sensors deployed there will check the level of impurities and the presence of such bacteria that are dangerous for both human beings and animals. According to the level of impurities even this can be detected that the water is fit for agriculture, animal consumption or for the human consumption.[3] If the water is unfit for human and animal consumption but fit for agricultural uses then it can be detected for the agricultural processes. Since there is running water in the river continuous monitoring of it tells the right time when the water is completely fit for human consumption.



EOS -300 Water Sensors[3]

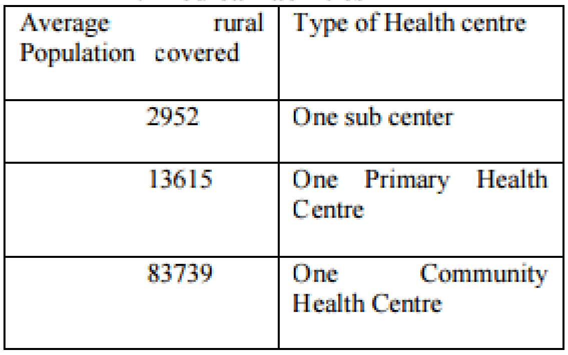
2.4 Arvind Rehalia , Dr SVAV Prasad : eHealth model for Himachal Pradesh (IJMER)

This paper presents a electronic health set for HP. This can be a good option for providing medical facilities in rural area of the HP.As internet plays a main role in the infrastructure of India. Internet can be used to enhance medical facilities in the rural India. This paper proposes a complete model for HP using internet. This model can be call as e-health model for HP.

Himachal Pradesh

Himachal is a state in the north of India having area of 55673 km square. Having total population of 6856509. Having Census Villages 20690 and Inhabited Villages 17495 Rural Population 6167805 and urban population 6,88,704. There are twelve district of HP. They are Kangra, Hamirpur, Mandi, Bilaspur, Una, Chamba, Lahul and Spiti, Sirmaur, Kinnaur, Kullu, Solan and Shimla. There are two Medical colleges in HP: RPMC Tanda, Kangra and IGMC Shimla1. Himachal Pradesh Himachal is a state in the north of India having area of 55673 km square. Having total population of 6856509. Having Census Villages 20690 and Inhabited Villages 17495 Rural Population 6167805 and urban population 6,88,704. [3]There are twelve district of HP. They are Kangra, Hamirpur, Mandi, Bilaspur, Una, Chamba, Lahul and Spiti, Sirmaur, Kinnaur, Kullu, Solan and Shimla. There are two Medical colleges in HP: RPMC Tanda, Kangra and IGMC Shimla

2. Medical facilities in HP

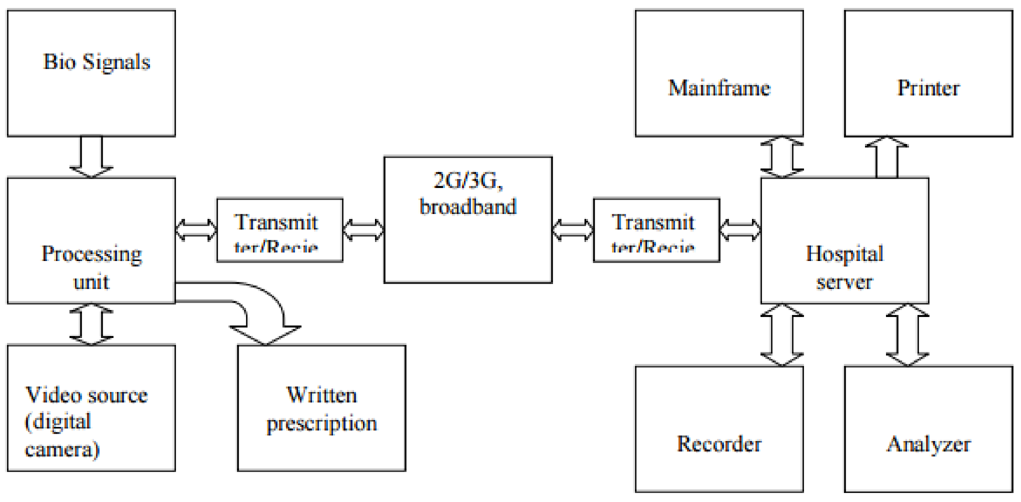


Average rural Population covered	Type of Health centre
2952	One sub center
13615	One Primary Health Centre
83739	One Community Health Centre

Working Model

There will be two groups group-1 and group-11. These groups are divided on the basis of distance from the medical college to the concerned district. Group -1 is connected with the medical college kangra and the group -11 is connected to IGMC Shimla as it is these district are near to shimla. Group –I (Medical college kangra)Kangra,Chamba,Hamirpur,Mandi,Una and Kullu Group-II(IGMC Shimla) Shimla,Sirmour,Kinnuaur,Bilaspur,Laul and spiti, and Solan.[4]

Block diagram of the system



Bio signals: Bio signals are the signals received from the portable biomedical instruments. As we can have ECG, EEG, BP and temperature measurement. We can have this measurements on the computer using various interface. [4]

Video source In case of 3G we can transmit video signal also which can help the doctor in proper diagnosis. In other case we can send the clip of the patient using MMS.

Printer Printer is used on both sides at the remote end and the hospital end for the printing purposes.

Analyzer Analyzer here is general physician who has to diagnose anemia and give the prescription according the patient conditions.

Main frame Mainframes are mainly a powerful computers used by corporate and governmental organizations for big applications, bulk data processing. Here it is used for the storage of data i.e electronic health records.

EPR

An electronic health record (EHR) which is sometimes also called as electronic patient record (EPR) is a concept which can be defined as an organised and systematic collection of patient’s health information electronically. It is a record in digital format and is capable of being shared across different health care units with the help of network connected information systems. These records generally include a whole range of data including their basic details or demographics of patient, their medical history, medication status, allergies, laboratory test results, radiology images, vital signs or symptoms of the disease. These records will be available to the doctors on their PCs. EPR will be available in the hospital server room.[4]

Reliability

Reliability is the main requirement of the system. There is lot of error in the measurement system. There is error due human interference. We will make the data more reliable. We will make the system more secure so that nobody can easily access or disturb the data. The data will remain secure. There will be locks for the security of the whole system. For checking the reliability model is required. Model will be developed according the equipment / instruments used in the whole system. [4]

Security of data

Security of data is the demand of time. Our data should be very much secured .various layers and types of information security control are appropriate to databases, including Access control , Auditing , Authentication , Encryption ,Integrity controls, Backups, Application security[4].

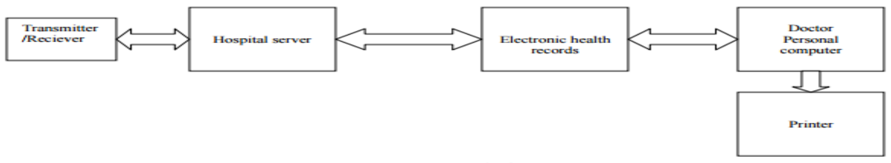


Fig.3

Village end

In village a room and a technical operator will be required. That room is installed with computer. That computer will be interfaced with ECG,EEG, BP machine, printer, webcam, speakers. That computer will be provided internet facilities through landline phone. For making this system more robust we will be having manually operated portable biomedical instruments in this room. For internet we will having wireless internet connection (dongle) in case of broadband landline system failure. In case of failure in the interface of instruments with computer operator can mail the values of patient reading using manual PBI. [4]

Hospital end

In the hospital a operator will be available in every department. That operator and doctor will receive the mail of the patient operator will upgrade database and the doctor will reply the mail to the operator and the patient. Operator will again upgrade the database. This database will be available on the doctors PC. Doctor can refer the database. The administrator / operator can only make changes in the database.[4]

<u>**Whole process**</u>

Patient:

Patient has to go to the hospital. He has to fill the form in the hospital. That form is given to the receptionist. Receptionist will give a slip to the patient on which following information will be given.

Registration: Name :Raj Kumar

Mail id: raj.kumar@ehphealth.com

Password: raju Patient id: 13572

Doctor Id: ca.01tanda@ehphealth.com

operator :op.02tanda@ehphealth.com

The doctor id will vary according to the disease which the patient is suffering from. ehphealth will be the server address hospital have to own. [4]

After his checkup patient does not need to come back to hospital for repeated checkups. He has to go to his village center. He has to show his slip to the operator. Then village operator will mail the current condition of the patient to the doctor and the concerned departmental operator through the patients mail id. Doctor will study his mail and will reply his suggestion to both patient and the departmental operator through mail. Departmental operator will add all these suggestions to the data base. Village operator can send patients ECG,EEG ,BP, Temperature etc record to doctor if required by doctor.[4]

Merits

- This system will help us to provide medical facility in every village of the state.
- This system will provide job to opportunity to local people
- This system will help in checking the number of patients of particular disease.
- Patient does not need to visit doctor again and again.

Demerits

- Used only for pre and post-operative treatments
- If data is wrong then prescription will be wrong
- Some technical knowledge is required to the patient

Future scope

This system can be used to make world free from diseases. It can be used to remove epidemics from the society. This concept can be used for developing for improving health sector .This concept can be further converted into virtual hospitals[3].

CHAPTER-3
SYSTEM DEVELOPMENT

3.1 REQUIREMENTS

3.1.1 Functional Requirements

a) Arrangement of Sensors

- The system has allotted 1 environmental for each farm.
- The system has allotted a human sensor to each cow.

b) Collecting Data

- The system will provide random values to all the sensors.
- These values shall be compared to the given standards.

c) Detect abnormal behavior

- The system will detect abnormal behavior and report it to the mainframe.

d) Prediction

- The system will predict the future values given on average values over a specified period.

3.1.2 Non-Functional Requirements

a) Security

- The system must guarantee the security of the network information.

b) Concurrency

- The system should be able to handle multiple operations concurrently.

c) Availability

- The system must be available 24/7 for the network.

d) Compatibility

- The system must be easy to use and compatible with user system.

3.1.3 System Requirements

We have implemented our project on IDE Eclipse Kepler. Eclipse runs on today's most popular operating systems, including Windows XP, Linux, and Mac OS X. It requires Java to run, so if you don't already have Java installed on your machine, you must first install a recent version. Mac OS X has Java preinstalled. See the following table for the minimum and recommended system requirements.

Requirement	**Minimum**	**Recommended**
Java version	1.4.0	5.0 or greater
Memory	512 MB	1 GB or more
Free Disk Space	300 MB	1 GB or more
Processor Speed	800 MHz	1.5 GHz or faster

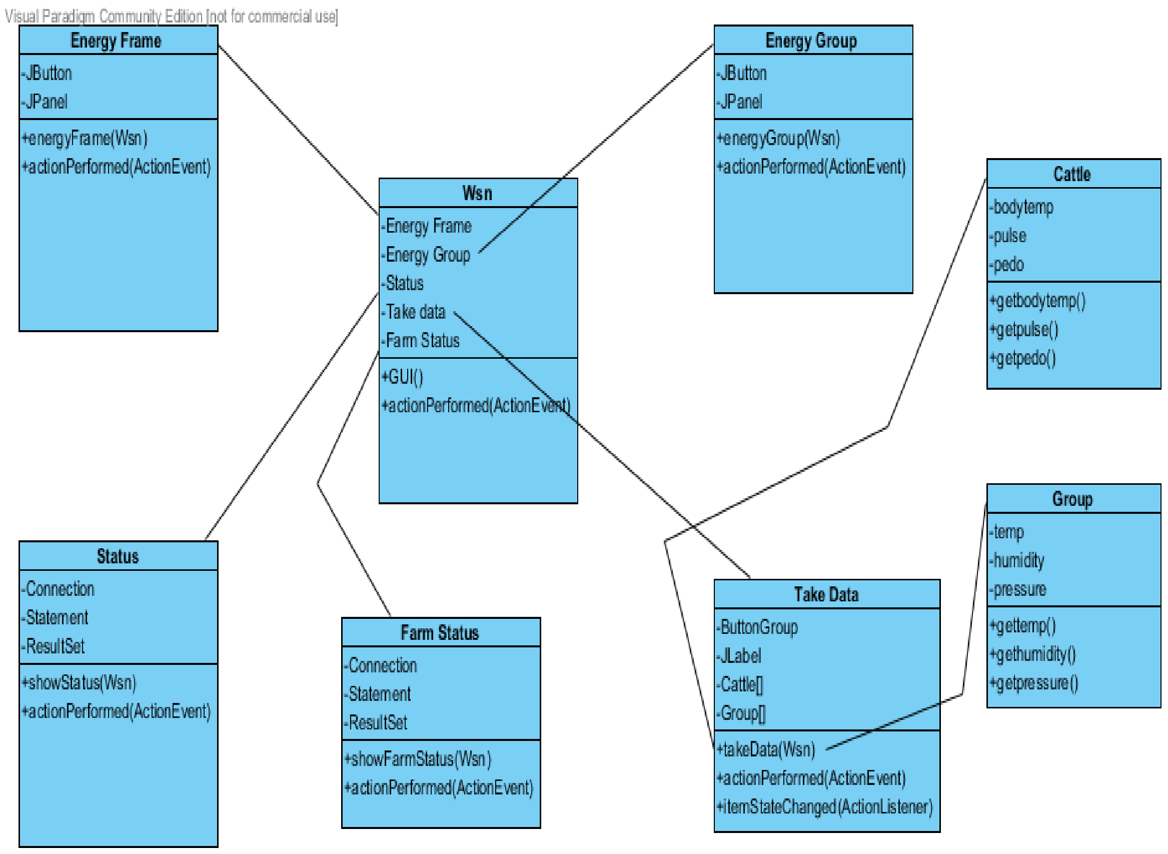
Table 1: System Requirements

3.2 PROCESS MODEL

For this project, we would suggest Spiral Model of software development process models which is an evolutionary software process model that couples the iterative nature of prototyping with controlled and systematic aspects of the waterfall model. A spiral model is divided into a number of framework activities also called task regions. A spiral model contains four task regions:

- **Determine Objectives**: This phase involves a good communication with the customer to well define the objectives or requirements for the software.
- **Risk Analysis**: This phase involves analyzing each aspect of software development in order to find every possible area that may involve risk while developing the software and thus designing modules accordingly.
- **Development:** This phase involves developing various diagrams and developing software according to the specification and thus testing the modules for their correct functioning.
- **Planning:** This phase involves planning for the next increment so that the increment may fit into already developed prototype and thus function according to the need.

3.3 DESIGN



3.4 IMPLEMENTATION

For implementation we have selected Java as our platform and we have implemented this using Eclipse *Kepler*. We have implemented this project using *Swings* because of various GUI features available in it. The code is given in Appendix.

3.4.1 Platform

A *platform* is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

- The *Java Virtual Machine* (Java VM)

- The *Java Application Programming Interface* (Java API)

You've already been introduced to the Java VM. It's the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as *packages*. The next section, What Can Java Technology Do?, highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.

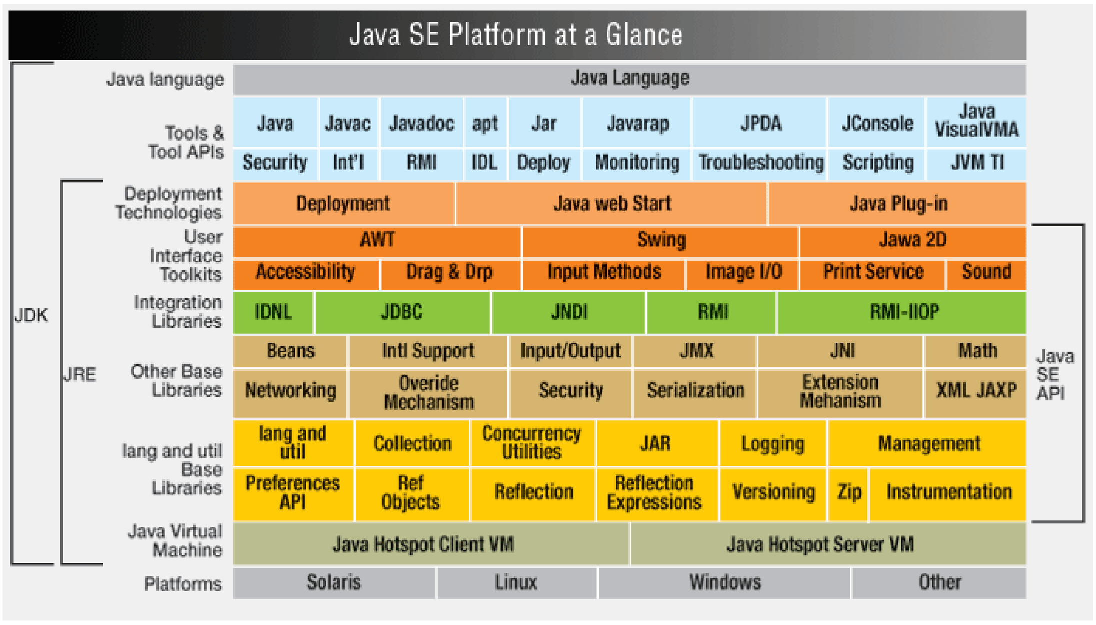


Figure: Java Platform

Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time bytecode compilers can bring performance close to that of native code without threatening portability.

3.4.2 Features of Java

The most common types of programs written in the Java programming language are *applets* and *applications*. If you've surfed the Web, you're probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

An application is a standalone program that runs directly on the Java platform. A special kind of application known as a *server* serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a *servlet*. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provide a wide range of functionality. Every full implementation of the Java platform gives you the following features:

- **The essentials**: Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.

- **Applets**: The set of conventions used by applets.

- **Networking**: URLs, TCP (Transmission Control Protocol), UDP (User Data gram Protocol) sockets, and IP (Internet Protocol) addresses.

- **Internationalization**: Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.

- **Security**: Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
- **Software components**: Known as JavaBeansTM, can plug into existing component architectures.

- **Object serialization**: Allows lightweight persistence and communication via Remote Method Invocation (RMI).

- **Java Database Connectivity (JDBCTM)**: Provides uniform access to a wide range of relational databases.

The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.

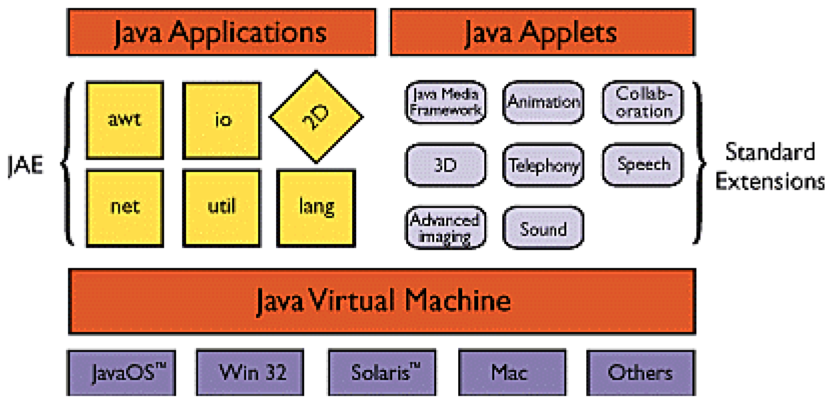


Figure: Java Architecture

JAVA Multithreading

Multithreading in java is a process of executing multiple threads simultaneously.Thread is basically a lightweight sub-process, a smallest unit of processing. Multiprocessing and multithreading, both are used to achieve multitasking.

But we use multithreading than multiprocessing because threads share a common memory area. They don't allocate separate memory area so saves memory, and context-switching between the threads takes less time than process.

Java Multithreading is mostly used in games, animation etc.

Advantage of Java Multithreading

1) It **doesn't block the user** because threads are independent and you can perform multiple operations at same time.

2) You **can perform many operations together so it saves time**.

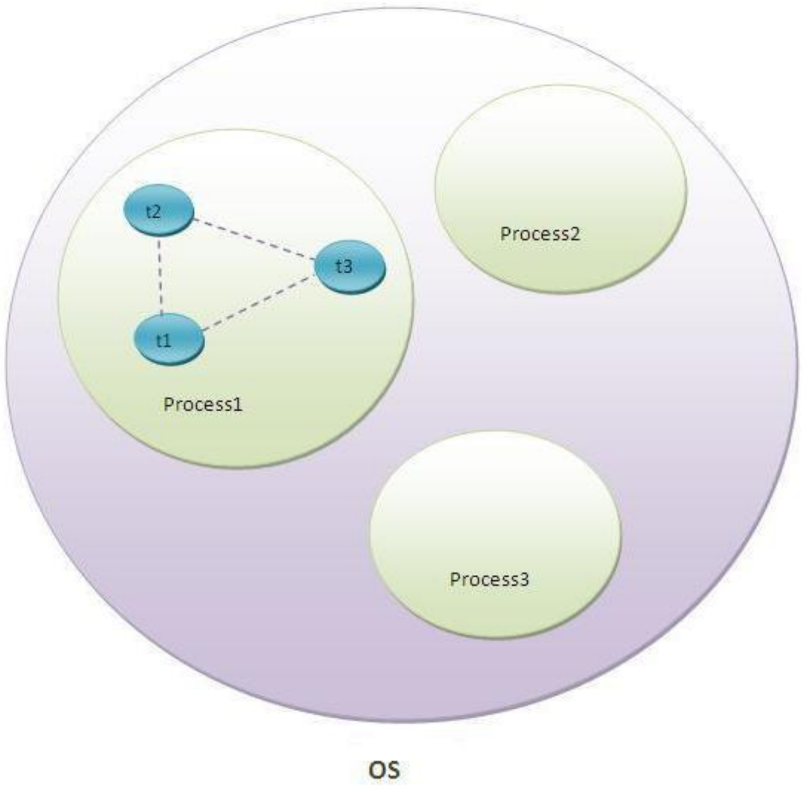
3) Threads are **independent** so it doesn't affect other threads if exception occur in a single thread.

What is Thread in java

A thread is a lightweight sub process, a smallest unit of processing. It is a separate path of execution.

Threads are independent, if there occurs exception in one thread, it doesn't affect other threads. It shares a common memory area.

As shown in the above figure, thread is executed inside the process. There is context-switching between the threads. There can be multiple processes inside the OS and one process can have multiple threads.



3.4.5 Java Swings

Swing is the principal GUI toolkit for the Java programming language. It is a part of the JFC (Java Foundation Classes), which is an API for providing a graphical user interface for Java programs. It is completely written in Java.

Swing library is an official Java GUI toolkit released by Sun Microsystems. It is used to create Graphical user interfaces with Java.

The main characteristics of the Swing toolkit:

- platform independent
- customizable
- extensible
- configurable
- lightweight

The Swing API has 18 public packages:

- javax.accessibility
- javax.swing
- javax.swing.border
- javax.swing.colorchooser
- javax.swing.event
- javax.swing.filechooser
- javax.swing.plaf
- javax.swing.plaf.basic
- javax.swing.plaf.metal
- javax.swing.plaf.multi
- javax.swing.plaf.synth
- javax.swing.table
- javax.swing.text
- javax.swing.text.html
- javax.swing.text.html.parser
- javax.swing.text.rtf
- javax.swing.tree
- javax.swing.undo

Swing is an advanced GUI toolkit. It has a rich set of widgets. From basic widgets like buttons, labels, scrollbars to advanced widgets like trees and tables. Swing itself is written in Java.

Swing is a part of JFC, Java Foundation Classes. It is a collection of packages for creating full featured desktop applications. JFC consists of AWT, Swing, Accessibility, Java 2D, and Drag and Drop. Swing was released in 1997 with JDK 1.2. It is a mature toolkit.

The Java platform has Java2D library, which enables developers to create advanced 2D graphics and imaging.

There are basically two types of widget toolkits.

- Lightweight
- Heavyweight

A heavyweight toolkit uses OS's API to draw the widgets. For example Borland's VCL is a heavyweight toolkit. It depends on WIN32 API, the built in Windows application programming interface. On Unix systems, we have GTK+ toolkit, which is built on top of X11 library. Swing is a lightweight toolkit. It paints its own widgets. Similarly does the Qt4 toolkit.

SWT library

There is also another GUI library for the Java programming language. It is called SWT. The Standard widget toolkit. The SWT library was initially developed by the IBM coorporation. Now it is an open source project maintained by the Eclipse community. The SWT is an example of a heavyweight toolkit. It lets the underlying OS to create GUI. SWT uses the Java native interface to do the job.

3.4.6 Databases and the Web

Databases are at the root of all business computing today. At some point, you are going to want to integrate a company database with your Web site. On the other hand, perhaps when you have a large Web site, you will want to create a database that will let clients search the text of your HTML files.
There are several ways to achieve database and Web site integration. Let’s concentrate on JDBC.

JAVA DB

DB is Oracle's supported distribution of the open source Apache Derby database. Its ease of use, standards compliance, full feature set, and small footprint make it the ideal database for Java developers. Java DB is written in the Java programming language, providing "write once, run anywhere" portability. It can be embedded in Java applications, requiring zero administration by the developer or user. It can also be used in client server mode. Java DB is fully transactional and provides a standard SQL interface

The Java Database Connectivity (JDBC)

The Java Database Connectivity (JDBC) API is the industry standard for database-independent connectivity between the Java programming language and a wide range of databases – SQL databases and other tabular data sources, such as spreadsheets or flat files. The JDBC API provides a call-level API for SQL-based database access.

JDBC technology allows you to use the Java programming language to exploit "Write Once, Run Anywhere" capabilities for applications that require access to enterprise data. With a JDBC technology-enabled driver, you can connect all corporate data even in a heterogeneous environment.

CHAPTER-4
PERFORMANCE ANALYSIS

4.1 ASSUMPTIONS

Following are the assumptions based on which the proposed algorithm has been implemented:

- The network is static.
- Deployment of nodes is predefined.
- Algorithm will cover the experimental area of 500X500 unit dimensions.
- Number of groups in which cattle are divided – 9.
- Number of cattle in each group – 4
- Total no of cattle under observation – 36
- Environmental sensor used for monitoring surrounding conditions.
- Physical sensors used for monitoring body conditions.

4.2 ANALYSIS

In this section we will show the performance analysis of different protocols used for different steps in our proposed solution and their topology:

Protocol For	**Flat Topology**	**Cluster Based**	**Tree Based**	**Chain Based**
Data Gathering	Yes	Yes	Yes	Yes
Target Tracking	Yes	Yes	Yes	Yes
Routing	Yes	Yes	Yes	Yes
Data Aggregation	Yes	Yes	Yes	Yes
Data Dissemination	Yes	Yes	Yes	Yes
Synchronization	Yes	Yes	Yes	No

Different protocols and their corresponding topology

Comparison of Different Topology:

1. **Based on Energy Efficiency:**

Communication is the most energy intensive activity performed by the sensor nodes, and hence the WSN topology and communication protocols can play a significant role in the energy efficiency and lifetime of the WSN.

Energy consumption of the sensor nodes of a WSN should be evenly distributed. As a result, those nodes in cluster and tree topologies deplete their energy faster than other nodes, and thereby disconnecting the base station from the whole WSN, which might still have adequate resources and infrastructure. This is the well-known self-induced black hole effect. Simulations shows that nodes closest to the base station are the ones to die out first for flat mesh routing, whereas nodes farthest from the base station are the ones to die out first for direct transmission.

In-network processing, which is primarily based on the motivation that typically computation is more energy-efficient than communication, is one of the key mechanisms with the potential to considerably improve the energy efficiency of WSNs. Simulations have shown that it typically requires around 100 to 1000 times more energy to transmit a bit than to execute an instruction. Possibilities for in-network processing in WSN include aggregation and compression, which exploit spatial and temporal correlation in the sensed data, for performing local compression to reduce global communication to base station by reducing the overhead of packet headers, by compressing the payload, and by reducing the probability of packet collisions.

2. **Based on Reliability:**

The WSN reliability can be studied for three different scopes of data delivery, collectively known as the infrastructure communication: (a) users send their interest to a single sensor node, (b) users send their interest to a subset of nodes in a sub-area and the message needs to be delivered to all sensors in the particular group, and (c) users send their interest to the entire sensor network and the message needs to be delivered to all sensors in the network. There exists another scope of message delivery, known as application communication, in which it is sufficient that the message from sink is reliably delivered to only a group of sensor nodes that together cover the entire sensor field or the intended area of observation. This is different from the delivery to all sensors in the infrastructure communication, due to the typical redundant deployment of sensors.

Besides, if certain environmental or architectural conditions results in poor reliability, it is difficult or impossible to adapt a point-to-point network like the network with a cluster topology to increase reliability. In contrast, the WSN reliability can be improved by redeploying redundant nodes in the affected area.

The tree topology has the lowest reliability due to the use of only a single direct link between nodes at successive levels in the hierarchy. Chain-oriented topology offers better reliability than tree-based topologies as in the same hierarchy level it uses multi-hop communication from source to sink. Clustered hierarchical topology is a compromise between the two extremes. It is better than tree/chain-oriented topology, as it maintains multi-hop paths, while it has lower reliability than flat topology because each communication between nodes at different clusters must route through affiliated cluster heads.

3. Based on Scalability and Self-Organization:

Self-organization helps in maximizing the network lifetime. Nevertheless, self-organization should be kept in perspective with energy cost and speed. Sometimes letting a WSN to kill its nodes may be more energy efficient than trying to revive it. Self-configuration time involves fault identification, fault localization and fault recovery phases.

In flat topology, high number of sensor nodes increases load on the base station, which results in increased power consumption and complexity. In addition, as node density increases, the increase in collisions greatly degrades performance. Further, not all nodes have enough transmission range or the line-of-sight communication with the base station. It is difficult to scale flat WSN to more than a few nodes.

For tree-based topologies, self-configuration and scalability are limited up to a certain number of depths of the tree. Therefore, in practice, tree-based topology works well for medium sized networks, but has scalability limitations that degrade performance for larger or densely deployed WSN.

The scalability and self-organization issues of chain-based topology primarily depend on the number of chains in the network. Chain-oriented topology with a single chain (PEGASIS) has the same limitations that the tree-based topologies have. Multiple chain-based topology and clustered hierarchical topologies improve the scalability of the flat networks by assigning leaders and/or cluster heads to manage the local neighbourhood of sensor nodes.
Besides, for scalability, the addressing structures of WSN are likely to be quite different; for example, geographic, data-centric, or address-free structures. Distributed and/or probabilistic assignments of addresses are only unique in a two-hop neighborhood.

4. <u>Based on Data Latency:</u>

The WSN traffic, which is characterized by the interaction with the environment or generated in response to certain events, is likely to be very different from human-driven forms of networks. A typical consequence is that WSNs are expected to exhibit very low data rates over a large time scale, but can have very burst traffic upon the occurrence of the certain events.

A single-hop-to-sink structure would have the least data latency because there is no delay due to buffering at routers along the path. However, it is not scalable and there may be more loss due to collisions as the network density increases. Flat topology networks have higher data latency than the single-hop-to-sink structure but lower data loss because keeping the transmission power lower reduces the packet collision rates.

In hierarchical tree topology, as the data moves from the lower level to a higher level, it moves a greater distance, thus reducing the travel time and data latency. However, as the distance cluster levels increases, the energy dissipation, which is proportional to square of distance, increases.

5. <u>Based on Overhead and Efficiency:</u>

Flat topology produces the maximum number of packets for routing. In a flat topology, because of flooding, a node can receive multiple copies of same data from different nodes. On the other hand, in cluster-based topology, the cluster heads receive data from all members of the cluster. In tree topology, a parent node receives data from its children node(s). However, in a chain-based topology, a node in a chain receives data from only one node. Thus, in terms of communication overhead, chain-based topology is the best, tree-based topology is the second best, cluster-based topology is good, and flat topology is bad.

In terms of control overhead, which measures the ratio between control and data messages in the network, flat topology is the best because it does not need to maintain any structure. Thus, this topology does not need to disseminate topology control messages. Tree-based topology, on the other hand, has the largest number of control message overhead to maintain the tree structure. Cluster-based and chain-based topologies also have control message overheads, but much less than that of the tree-based topology.

Comparison Table Analysis

Table summarizes the comparative analysis of the four topologies based on the assumptions. The topologies were compared with ten performance metrics, namely total energy consumption, energy distribution, load distribution, redundant communication, data reliability, scalability, latency, network connectedness, lifetime, and topology management overhead. Finally, the points for each evaluation metrics of each topology are added. The totals, which indicate the overall performance, for each topology are shown at the bottom most row of the table.

In our point system, 4 means excellent, 3 means good, 2 means fair and 1 means poor. Depending on the design structure of a topology, some fields of the table have a range of numbers presented as x to y. For example, with regard to latency, single chain-based topology shows fair results, whereas multi-chain topology shows excellent results. Thus for latency, the chain-based topology received 2 to 4. These points used here are entirely relative.

The table shows that the chain-oriented topology scores the highest among four topologies, whereas the flat topology scores the lowest. The cluster-based topology performs better than the tree-based topology but not as efficiently as the chain-based topology. Nevertheless, there are some areas in the chain-based topologies to which special attentions should be paid by the designer to make the topology more efficient.

Evaluation Metric	**Flat**	**Cluster-Based**	**Tree-Based**	**Chain-Based**
Total Energy Consumption	1	3	2	4
Energy Distribution	1	3	2	4
Load Distribution	3	3	3	4
Redundant Communication	1	4	4	4

Data Reliability	4	3	3	2 to 3
Scalability	2	4	3	2 to 4
Latency	4	3	3	2 to 4
Network Connectedness	1	3	3	3
Lifetime	2	3	3	4
Topology Management overhead	4	3	2	4
Overall Scores	23	32	28	32 to 37

Table: Comparison of Different Topology

4.3 Sample Output

- Total number of nodes: 36
- Number of Group nodes: 9
- Number of Cattle nodes per group: 4

S.No.	Sensor Tested	Output	Action Required
1	Group 1	• Temperature: 37.4 C • Humidity: 60% • Atmospheric Pressure: 30.3 Hg	No Action Required.
2	Group 1-Catlle 1	• Body Temperature: 37.4 C • Pulse Rate : 46.2 pm • Pedometer: 3.5 km	No action required.
3	Group 1-Catlle 3	• Body Temperature: 38.2 C • Pulse Rate : 45.1 pm • Pedometer: 3.6km	No action required
4	Group 2-Catlle 4	• Body Temperature: 48.2 C • Pulse Rate : 57.1 pm • Pedometer: 1.5km	Warning Message sent to Farm Owner.
5	Group 7-Catlle 2	• Body Temperature: 36.2 C • Pulse Rate : 43.3 pm • Pedometer: 4.6km	No action required

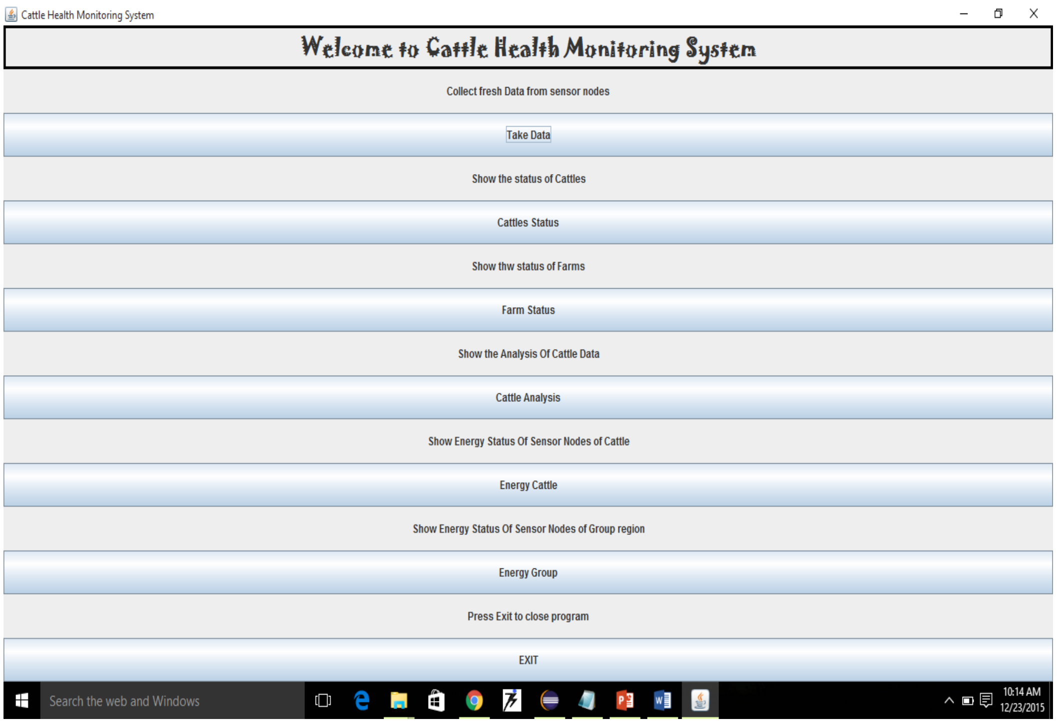
6	Group 5-Catlle 3	• Body Temperature: 50.2 C • Pulse Rate : 62.1 pm • Pedometer: 0.8km	Warning Message sent to Farm Owner.
7	Group 6	• Temperature: 36.2 C • Humidity : 70% • Atmospheric Pressure: 32.2 Hg	No action required
8	Group 9	• Temperature: 47.2 C • Humidity:90% • Atmospheric Pressure: 37.2 Hg	Warning Message sent to Farm Owner.

4.4 RESULTS

The algorithm successfully processed of all the nodes either cattle node or group node for the test cases. As the data is collected from all the nodes and has been processed by central hub and response has been generated based on criteria different parameters gathered from nodes have met.

In addition to this, we can also get energy level of each node by testing for energy level of farm nodes or cattle nodes. The nodes with energy level greater than 20 are in safe range, while nodes with less than 10 are in red zone and needs attention. We can also test for cattle or group by taking fresh data at our own choice whenever we need them otherwise it will take data from sensors at regular intervals.

4.5 SNAPSHOTS


Cattle Health Monitoring System

Welcome to Cattle Health Monitoring System

Collect fresh Data from sensor nodes

Take Data

Show the status of Cattles

Cattles Status

Show thw status of Farms

Farm Status

Show the Analysis Of Cattle Data

Cattle Analysis

Show Energy Status Of Sensor Nodes of Cattle

Energy Cattle

Show Energy Status Of Sensor Nodes of Group region

Energy Group

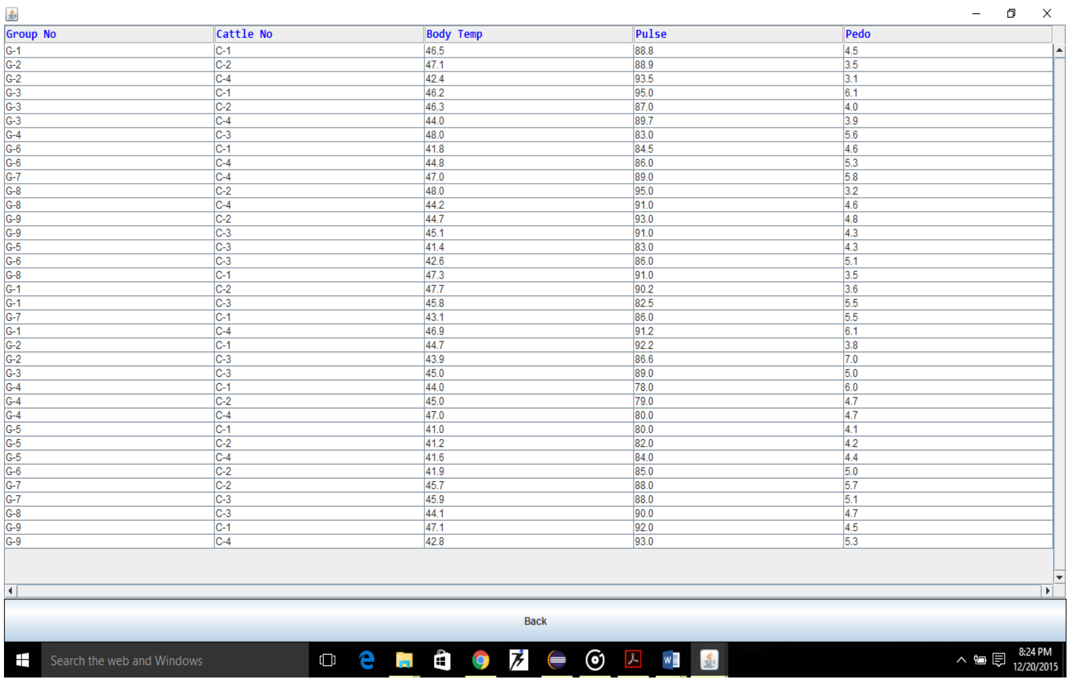
Press Exit to close program

EXIT

Search the web and Windows

10:14 AM
12/23/2015

Figure: Menu



Group No	Cattle No	Body Temp	Pulse	Pedo
G-1	C-1	46.5	88.8	4.5
G-2	C-2	47.1	88.9	3.5
G-2	C-4	42.4	93.5	3.1
G-3	C-1	46.2	95.0	6.1
G-3	C-2	46.3	87.0	4.0
G-3	C-4	44.0	89.7	3.9
G-4	C-3	48.0	83.0	5.6
G-6	C-1	41.8	84.5	4.6
G-6	C-4	44.8	86.0	5.3
G-7	C-4	47.0	89.0	5.8
G-8	C-2	48.0	95.0	3.2
G-8	C-4	44.2	91.0	4.6
G-9	C-2	44.7	93.0	4.8
G-9	C-3	45.1	91.0	4.3
G-5	C-3	41.4	83.0	4.3
G-6	C-3	42.6	86.0	5.1
G-8	C-1	47.3	91.0	3.5
G-1	C-2	47.7	90.2	3.6
G-1	C-3	45.8	82.5	5.5
G-7	C-1	43.1	86.0	5.5
G-1	C-4	46.9	91.2	6.1
G-2	C-1	44.7	92.2	3.8
G-2	C-3	43.9	86.6	7.0
G-3	C-3	45.0	89.0	5.0
G-4	C-1	44.0	78.0	6.0
G-4	C-2	45.0	79.0	4.7
G-4	C-4	47.0	80.0	4.7
G-5	C-1	41.0	80.0	4.1
G-5	C-2	41.2	82.0	4.2
G-5	C-4	41.6	84.0	4.4
G-6	C-2	41.9	85.0	5.0
G-7	C-2	45.7	88.0	5.7
G-7	C-3	45.9	88.0	5.1
G-8	C-3	44.1	90.0	4.7
G-9	C-1	47.1	92.0	4.5
G-9	C-4	42.8	93.0	5.3

Back

Search the web and Windows

8:24 PM
12/20/2015

Figure: Cattle Data

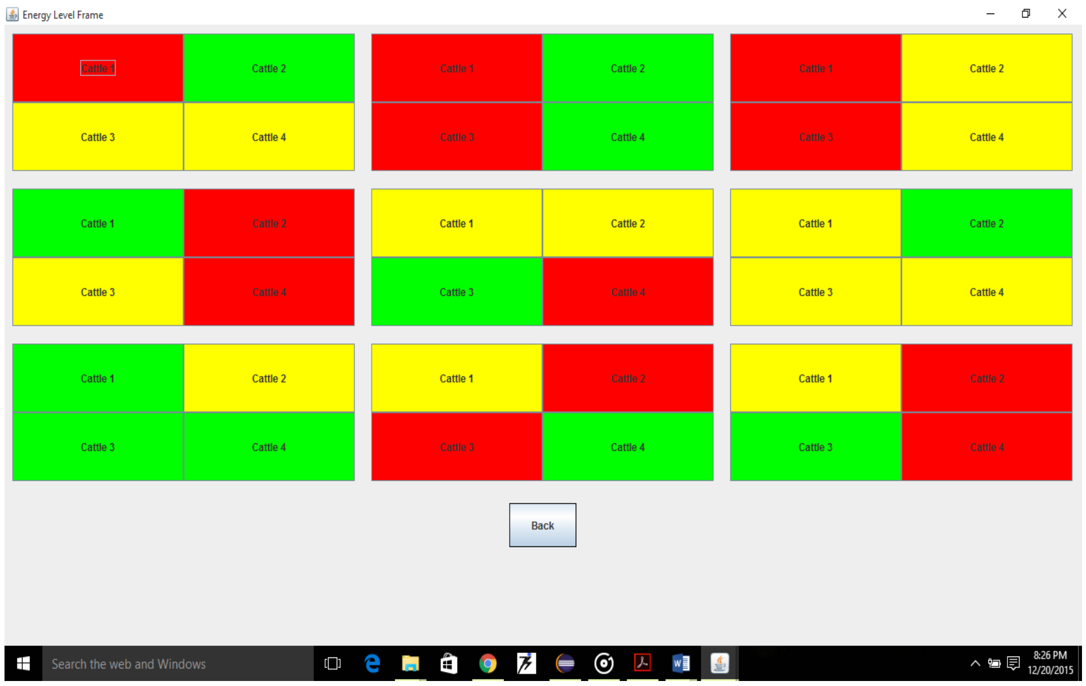


Figure: Cattle Nodes Energy Level

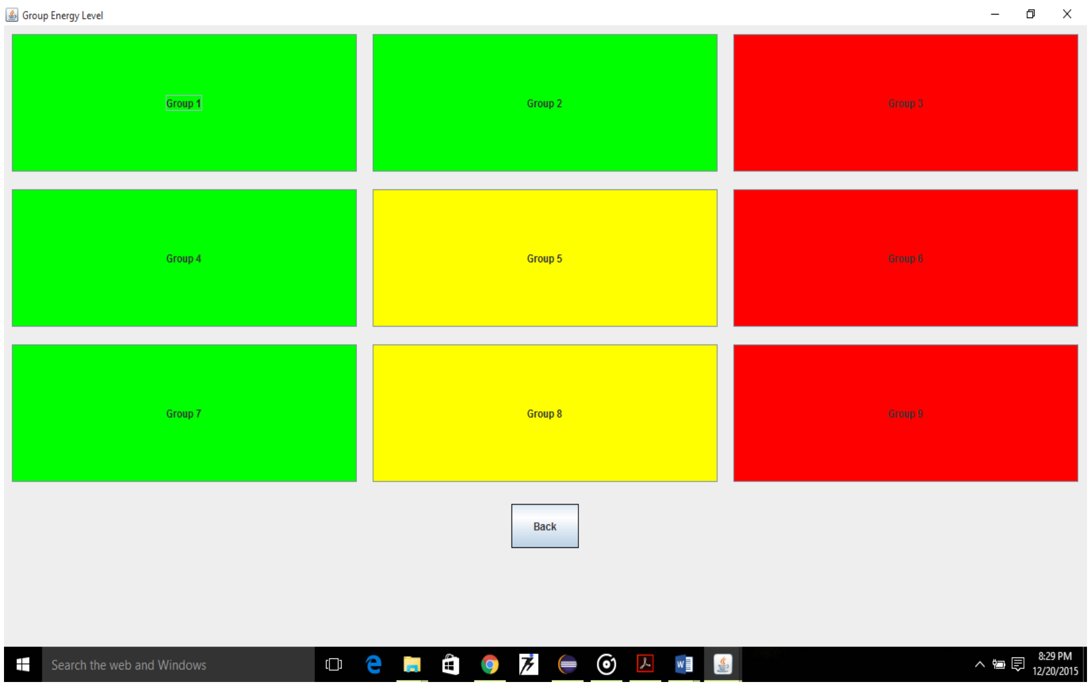


Figure: Farm Nodes Energy Level

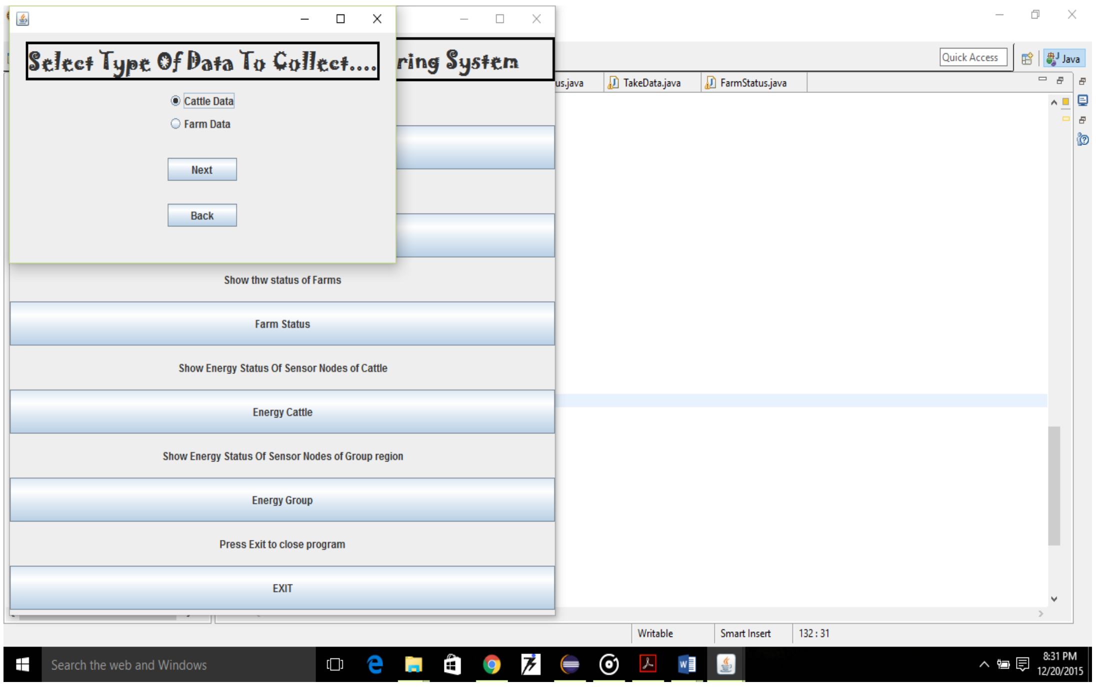
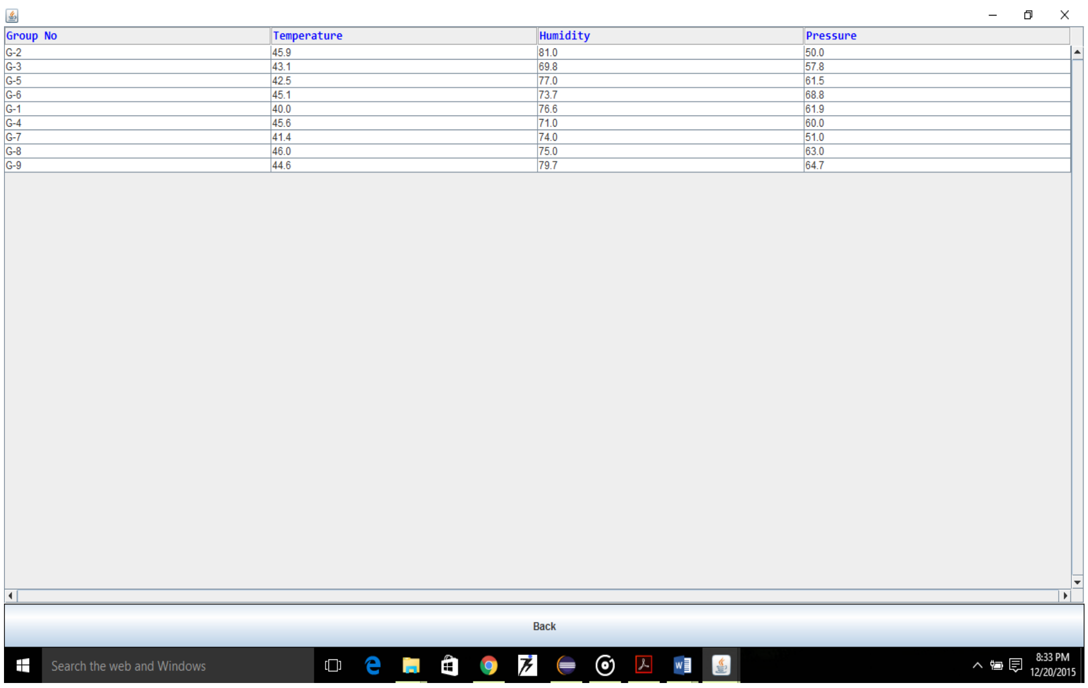

Select Type Of Data To Collect....
Cattle Data
Farm Data
Next
Back
ring System
Show thw status of Farms
Farm Status
Show Energy Status Of Sensor Nodes of Cattle
Energy Cattle
Show Energy Status Of Sensor Nodes of Group region
Energy Group
Press Exit to close program
EXIT

Figure: Fresh Data Collection



Group No	Temperature	Humidity	Pressure
G-2	45.9	81.0	50.0
G-3	43.1	69.8	57.8
G-5	42.5	77.0	61.5
G-6	45.1	73.7	68.8
G-1	40.0	76.6	61.9
G-4	45.6	71.0	60.0
G-7	41.4	74.0	51.0
G-8	46.0	75.0	63.0
G-9	44.6	79.7	64.7

Figure: Group Nodes Data

CHAPTER-5
CONCLUSION

5.1 CONCLUSIONS

With the advancement of technology the use of Wireless Sensor Network has increased manifolds. Almost all the major fields make the use of wireless sensors today whether it is agriculture, medical, environment, structural monitoring, rural connectivity, home security, airport security, disaster management, etc.

Wireless Sensor Network can easily be employed and can be handled by local people. Through this method we can have a prior idea of the health of cattle as well as an indication on what factors are responsible for illness and lead to lower productivity in cattle.

Rural people usually do not prefer visiting a doctor in case of some small infection or disease in cattle which can be dangerous, but this method will prove to be a step in the direction of improving the health conditions of cattle or at least preventing diseases and increase the productivity.

5.2 FUTURE SCOPE

The proposed algorithm predicts future behavior on basis of values given over a period of time but the next step could be to use Big Data so that we can get useful insights on the data that we receive from the nodes.

Also we can use Prediction analysis algorithm like SVM (Support Vector Machine) to predict future behavior of cattle.

Then first proposal is to monitor cattle health but we can extend it to monitor soil alkanity / acidity or landslide detection to prevent the local population to indulge in any practice which harms the environment or their health. Also we can deploy flood sensors to detect rise in river water level and issue warning to locals on time.

REFERENCES

[1]Ankit R. Bhavsar, Harshal A. Arolkar, Wireless Sensor Networks: A Possible Solution For Animal Health Issues In Rural Area Of Gujarat, 2 July 2012.

[2]Rashid Hussain, Purvi Mishra, Babita Sharma, Use of Wireless Sensor Network for Human and Animal Health Care during FloodWireless Sensor Networks, March 2013.

[3]Arvind Rehalia, SVAV Prasad, eHealth model for Himachal Pradesh, Mar-Apr 2012.

[4]Maneesha V. Ramesh, Sangeeth Kumar and P. Venkat Rangan, Wireless Sensor Network for Landslide Detection.

[5]Quazi Mamum, A Qualitative Comparison of Different Topologies for Wireless Sensor Networks, 5 November, 2012.

[6]Sunita Gupta, K.C. Roy, Comparison of different Energy Minimization Techniques in Wireless Sensor Network, August 2013.

[7]Shafiullah Khan, Wireless Sensor Networks- Current Status and Future trends.

[8]Anselemi B. Lukonge, Shubhi Kaijage, Ramadhani S. Sinde, Review Of Cattle Monitoring System Using Wireless Network, 5 May 2014.

[9]Kae Hsiang Kwong, Tsung Ta Wu, Hock Guan Goh, Ivan Andronovic, Wireless Sensor Networks in Agriculture: Cattle Monitoring For Farming Industries, 2009.

[10]James Foulkes, Peter Tucker, Chris Farnell, Livestock Management System.

[11]Y.Guo, P. Corke, T.Wark, D. Swain and G. Bishop-Hurley, Animal Behaviour Understanding using Wireless Sensor Networks.

[12]Mashoko Nkwari, Rimer, Paul, Cattle Monitoring System using wireless sensor network in order to prevent cattle rusting.

[13]Wu, T. and Goo, S.K. and Kwong, K.H. and Michie, C. and Andonovic, Wireless Sensor Network for Cattle Monitoring System, July 24 2009.

APPENDIX

SOURCE CODE

a) Wsn.java

```
package wsn;
//import packages

public class Wsn extends JFrame implements ActionListener
{
        private JPanel p1;
        private JButton b1,b2,b3,b4,b5,b6;
        private JLabel l1,l2,l3,l4,l5,l6,l7;
        EnergyFrame eframe;
        EnergyGroup eframegrp;
        Status status;
        TakeData takedata;
        FarmStatus farmstatus;
        public static Energy energyCattle[]=new Energy[36];
        public static Energy energyGroup[]=new Energy[9];
 public static void main(String args[])
 {
        for(int i=0;i<36;i++)
        {
                energyCattle[i]=new Energy();
        }
        for(int i=0;i<9;i++)
        {
                energyGroup[i]=new Energy();
        }
         Wsn mywsn=new Wsn();
         mywsn.Gui();

 }
 public void Gui()
 {
         JFrame frame=new JFrame("Cattle Health Monitoring System");
         frame.setLayout(new BorderLayout(3,1));

         p1=new JPanel();
         p1.setLayout(new GridLayout(13,1));

         Border border=BorderFactory.createLineBorder(Color.black,3);
         l1=new JLabel("Welcome to Cattle Health Monitoring System");
         l1.setVerticalAlignment(JLabel.CENTER);
```

```
l1.setHorizontalAlignment(JLabel.CENTER);
l1.setBorder(border);
l1.setPreferredSize(new Dimension(100,100));
Font font=new Font("Jokerman",Font.BOLD,25);
l1.setFont(font);
p1.add(l1);



b1=new JButton("Take Data");
b2=new JButton("Cattles Status");
b3=new JButton("Farm Status");
b4=new JButton("Energy Cattle");
b5=new JButton("Energy Group");
b6=new JButton("EXIT");

b1.addActionListener(this);
b2.addActionListener(this);
b3.addActionListener(this);
b4.addActionListener(this);
b5.addActionListener(this);
b6.addActionListener(this);

l2=new JLabel("Collect fresh Data from sensor nodes");
l2.setVerticalAlignment(JLabel.CENTER);
l2.setHorizontalAlignment(JLabel.CENTER);
p1.add(l2);
p1.add(b1);

l3=new JLabel("Show the status of Cattles");
l3.setVerticalAlignment(JLabel.CENTER);
l3.setHorizontalAlignment(JLabel.CENTER);
p1.add(l3);
p1.add(b2);

l4=new JLabel("Show thw status of Farms");
l4.setVerticalAlignment(JLabel.CENTER);
l4.setHorizontalAlignment(JLabel.CENTER);
p1.add(l4);
p1.add(b3);

l5=new JLabel("Show Energy Status Of Sensor Nodes of Cattle");
l5.setVerticalAlignment(JLabel.CENTER);
l5.setHorizontalAlignment(JLabel.CENTER);
```

```
            p1.add(l5);
            p1.add(b4);

            l6=new JLabel("Show Energy Status Of Sensor Nodes of Group region");
            l6.setVerticalAlignment(JLabel.CENTER);
            l6.setHorizontalAlignment(JLabel.CENTER);
            p1.add(l6);
            p1.add(b5);

            l7=new JLabel("Press Exit to close program");
            l7.setVerticalAlignment(JLabel.CENTER);
            l7.setHorizontalAlignment(JLabel.CENTER);
            p1.add(l7);
            p1.add(b6);



            frame.add(p1,BorderLayout.CENTER);
            frame.setSize(700,700);
            frame.setVisible(true);
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);




  }
@Override
public void actionPerformed(ActionEvent e) {

        if(e.getSource()==b1)
        {
                takedata=new TakeData();
                takedata.takeData(this);
                setVisible(false);
                dispose();

        }
        else if(e.getSource()==b2)
        {
                status=new Status();
                status.showStatus(this);
                setVisible(false);
                dispose();

        }
```

```
                else if(e.getSource()==b3)
                {
                        farmstatus=new FarmStatus();
                        farmstatus.showFarmStatus(this);
                        setVisible(false);
                        dispose();
                }
                else if(e.getSource()==b4)
                {
                        //this.dispose();
                  eframe=new EnergyFrame();
                  eframe.energyFrame(energyCattle,this);
                  setVisible(false);
                        dispose();
                }
                else if(e.getSource()==b5)
                {


                        eframegrp=new EnergyGroup();
                        eframegrp.energyGroup(energyGroup,this);
                        setVisible(false);
                        dispose();
                }
                else if(e.getSource()==b6)
                {
                        System.exit(0);
                }

        }
        }
```

b) EnergyGroup.java

```
package wsn;


import java.awt.Color;
//include header files


public class EnergyGroup extends JFrame implements ActionListener
{
  private JButton b[]=new JButton[10];
  private JPanel p1,p2,p3,p4,p5,p6,p7,p8,p9,p10;


        public void energyGroup(Energy energy[],Wsn wsn)
```

```
{
            JFrame frame=new JFrame("Group Energy Level");
            frame.setLayout(new GridLayout(4,3));

            int i=0;
   p1=new JPanel();
            p1.setLayout(new GridLayout(1,1));
            b[i]=new JButton("Group 1");
            p1.add(b[i]);
             b[i].addActionListener( new ActionListener( ) {
                 public void actionPerformed( ActionEvent e )
                 {
                   JOptionPane.showMessageDialog(null,"Energy Level is "+energy[0].getEnergyLevel());
                 }
               });
            p1.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
            i++;

            p2=new JPanel();
   p2.setLayout(new GridLayout(1,1));
            b[i]=new JButton("Group 2");
   p2.add(b[i]);
   b[i].addActionListener( new ActionListener( ) {
                 public void actionPerformed( ActionEvent e )
                 {
                   JOptionPane.showMessageDialog(null,"Energy Level is "+energy[1].getEnergyLevel());
                 }
               });
            p2.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
            i++;

   p3=new JPanel();
            p3.setLayout(new GridLayout(1,1));
            b[i]=new JButton("Group 3");
            p3.add(b[i]);
            b[i].addActionListener( new ActionListener( ) {
                 public void actionPerformed( ActionEvent e )
                 {
                   JOptionPane.showMessageDialog(null,"Energy Level is "+energy[2].getEnergyLevel());
                 }
               });
p3.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
i++;
```

```
p4=new JPanel();
            p4.setLayout(new GridLayout(1,1));
            b[i]=new JButton("Group 4");
            p4.add(b[i]);
            b[i].addActionListener( new ActionListener( ) {
                 public void actionPerformed( ActionEvent e )
                 {
                   JOptionPane.showMessageDialog(null,"Energy Level is "+energy[3].getEnergyLevel());
                 }
               });
p4.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
i++;

p5=new JPanel();
            p5.setLayout(new GridLayout(1,1));
            b[i]=new JButton("Group 5");
            p5.add(b[i]);
            b[i].addActionListener( new ActionListener( ) {
                 public void actionPerformed( ActionEvent e )
                 {
                   JOptionPane.showMessageDialog(null,"Energy Level is "+energy[4].getEnergyLevel());
                 }
               });
p5.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
i++;

p6=new JPanel();
            p6.setLayout(new GridLayout(1,1));
            b[i]=new JButton("Group 6");
            p6.add(b[i]);
            b[i].addActionListener( new ActionListener( ) {
                 public void actionPerformed( ActionEvent e )
                 {
                   JOptionPane.showMessageDialog(null,"Energy Level is "+energy[5].getEnergyLevel());
                 }
               });
p6.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
i++;

p7=new JPanel();
            p7.setLayout(new GridLayout(1,1));
            b[i]=new JButton("Group 7");
```

```
              p7.add(b[i]);
              b[i].addActionListener( new ActionListener( ) {
                    public void actionPerformed( ActionEvent e )
                    {
                      JOptionPane.showMessageDialog(null,"Energy Level is "+energy[6].getEnergyLevel());
                    }
                  });
p7.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
i++;

p8=new JPanel();
              p8.setLayout(new GridLayout(1,1));
              b[i]=new JButton("Group 8");
              p8.add(b[i]);
              b[i].addActionListener( new ActionListener( ) {
                    public void actionPerformed( ActionEvent e )
                    {
                      JOptionPane.showMessageDialog(null,"Energy Level is "+energy[7].getEnergyLevel());
                    }
                  });
p8.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
i++;

p9=new JPanel();
              p9.setLayout(new GridLayout(1,1));
              b[i]=new JButton("Group 9");
              p9.add(b[i]);
              b[i].addActionListener( new ActionListener( ) {
                    public void actionPerformed( ActionEvent e )
                    {
                      JOptionPane.showMessageDialog(null,"Energy Level is "+energy[8].getEnergyLevel());
                    }
                  });
p9.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
i++;

 p10=new JPanel();
              p10.setLayout(new FlowLayout());
              b[i]=new JButton("Back");
               Border border=BorderFactory.createLineBorder(Color.black);
               b[i].setVerticalAlignment(JLabel.CENTER);
                b[i].setHorizontalAlignment(JLabel.CENTER);
                b[i].setBorder(border);
```

```
                    b[i].setPreferredSize(new Dimension(85,50));
                   p10.add(b[i]);
                    b[i].addActionListener( new ActionListener( ) {
                        public void actionPerformed( ActionEvent e )
                        {


                         wsn.Gui();
                         setVisible(false);
                         dispose();
                        }
                      });
                   p10.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
                   i++;

for(int j=0;j<9;j++)
          {
                    if(energy[j].getEnergyLevel()<=10)
                    {
                              b[j].setBackground(Color.RED);
                              b[j].setContentAreaFilled(false);
          b[j].setOpaque(true);
                    }
                    else if(energy[j].getEnergyLevel()>10&&energy[j].getEnergyLevel()<=20)
                    {
                              b[j].setBackground(Color.YELLOW);
                              b[j].setContentAreaFilled(false);
          b[j].setOpaque(true);
                    }
                    else
                    {
                              b[j].setBackground(Color.GREEN);
                              b[j].setContentAreaFilled(false);
          b[j].setOpaque(true);
                    }
          }

           //add frames


                   frame.setSize(700,700);
                   frame.setVisible(true);
                   frame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
         }
```

c) TakeData.java

```
package wsn;

//import packages

public class TakeData extends JFrame implements ActionListener ,ItemListener
{
   private JLabel label;
   public ButtonGroup btngrp;
   public void takeData(Wsn wsn)
            {
                        JFrame frame=new JFrame();

                        frame.setLayout(new FlowLayout());
                        frame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
                        JPanel jp1=new JPanel();
                        jp1.setLayout(new FlowLayout());

                        Border border = BorderFactory.createLineBorder(Color.BLACK,3);
                        Font font=new Font("Jokerman",Font.BOLD,25);
                        label=new JLabel("Select Type Of Data To Collect....");
                        label.setVerticalAlignment(JLabel.CENTER);
                        label.setHorizontalAlignment(JLabel.CENTER);
                        label.setFont(font);
                        label.setBorder(border);
                        jp1.add(label);
                        //contentpane.add(label);
                        JPanel jp2=new JPanel();
                        jp2.setLayout(new GridLayout(6,1));
                btngrp=new ButtonGroup();
                        JRadioButton rb1=new JRadioButton("Cattle Data",true);
                        rb1.setActionCommand("Cattle Data");
                        btngrp.add(rb1);
                        jp2.add(rb1);



                        JRadioButton rb2=new JRadioButton("Farm Data");
                        rb2.setActionCommand("Farm Data");
                        btngrp.add(rb2);
                        jp2.add(rb2);



                        jp2.add(new JLabel());
```

```
                    //jp3.setLayout(new FlowLayout());
                    JButton next=new JButton("Next");
                    jp2.add(next);
                    next.addActionListener(new ActionListener(){
                            public void actionPerformed(ActionEvent e)
                            {
                            try
                            {
                            Class.forName("oracle.jdbc.driver.OracleDriver");


con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","Anshul","anshul");
                            if(selected.equals("Cattle Data"))
                            {

                                 Double temp,pulse,pedo;
                                 String
table[]={"CATTLESDATA","CATTLESDATA1","CATTLESDATA2","CATTLESDATA3","CATTLESDATA4"};
                                 for(int j=0;j<5;j++)
                                 {
                                  pstmt  =  con.prepareStatement("Insert  into  "  +table[j]+  "
values (?, ?, ?, ?)");

                                 for(int i=1;i<=36;i++)
                                 {
                                      temp=(double) rand.nextDouble()*(47-37)+37;
                                   pulse=(double) rand.nextDouble()*(35-15)+15;
                                  pedo=(double) rand.nextDouble()*(75-15)+15;
                                  pstmt.clearParameters();
                                  pstmt.setDouble(1,i);
                                  pstmt.setDouble(2,temp);
                                  pstmt.setDouble(3,pulse);
                                  pstmt.setDouble(4,pedo);
                                  pstmt.executeUpdate();


                                 }
                                 }

                            }
                            else if(selected.equals("Farm Data"))
                            {

                                  Double gtemp=2.1,humidity=3.1,pressure=4.4;
                                  pstmt = con.prepareStatement("Insert into FARMDATA values (?,
?, ?, ?)");
                                               //stmt=con.createStatement();
                                               Random rand1=new Random();
                               for(int i=1;i<=9;i++)
                                 {
                                      gtemp=(double) rand1.nextDouble()*37+1;
                                   humidity=(double) rand1.nextDouble()*20+62;
                                   pressure=(double) rand1.nextDouble()*10+26;
                                  pstmt.clearParameters();
                                  pstmt.setDouble(1,i);
                                  pstmt.setDouble(2,gtemp);
                                  pstmt.setDouble(3,humidity);
                                  pstmt.setDouble(4,pressure);
                                  pstmt.executeUpdate();
```

```
                              }

                        }
                          }
                         catch(Exception ex)
                               {
                                    System.out.println(ex);
                                    }              JOptionPane.showMessageDialog( null,

                     selected + " is collected." );
                          }
                    });
                    jp2.add(new JLabel());
                    JButton back=new JButton("Back");
                    jp2.add(back);
                     back.addActionListener( new ActionListener( ) {
                       public void actionPerformed( ActionEvent e )
                       {
                             setVisible(false);


                             wsn.Gui();
                             dispose();


                       }
                     });

                    frame.add(jp1);
                    frame.add(jp2);



                    frame.setSize(500,300);
                    frame.setVisible(true);



               }
          }
```

d) FarmStatus.java

```
    package wsn;

//import packages

public class FarmStatus extends JFrame{

       public  Connection con;
            public Statement stmt;
            public ResultSet rs=null;
            JButton b1;

      public void showFarmStatus(Wsn wsn) {

            JFrame frame=new JFrame();
            frame.setLayout(new BorderLayout());
            frame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);

            String query="Select * from FarmData";

            DefaultTableModel model=new DefaultTableModel();
            JTable table=new JTable(model);
            model.addColumn("Group No");
            model.addColumn("Temperature");
          model.addColumn("Humidity");
            model.addColumn("Pressure");

            table.getTableHeader().setDefaultRenderer(new SimpleHeaderRenderer());

            JScrollPane                                                scrollpane=new
JScrollPane(table,JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,
                    JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);

            try
      {
            Class.forName("oracle.jdbc.driver.OracleDriver");


con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","Anshul","anshul");

           stmt=con.createStatement();

           rs=stmt.executeQuery(query);
           int k=0;
                  while(rs.next())
                  {
                        String grp=rs.getString(1);
                        //String cattle=rs.getString(2);
                        Double temp=rs.getDouble(2);
                        Double humidity=rs.getDouble(3);
                        Double pressure=rs.getDouble(4);

                  //    Object row[]={grp,cattle,temp,pulse,pedo};
                        model.insertRow(k, new Object[]{grp,temp,humidity,pressure});
                        k++;
                  }
                  con.close();
         }
      catch(Exception e)
      {
            JOptionPane.showMessageDialog(null,"Cannot connect to Database");
      }
            b1=new JButton("Back");
             Border border=BorderFactory.createLineBorder(Color.black);
             b1.setVerticalAlignment(JLabel.CENTER);
              b1.setHorizontalAlignment(JLabel.CENTER);
              b1.setBorder(border);
              b1.setPreferredSize(new Dimension(50,50));
              b1.addActionListener( new ActionListener( ) {
                  public void actionPerformed( ActionEvent e )
                  {
                     setVisible(false);

                     wsn.Gui();
                     dispose();

                  }
                });
```

```
        //p2.add(b1);

    //frame.add(p1);
        //frame.add(p2);
        frame.add(scrollpane);
        frame.add(b1,BorderLayout.SOUTH);
        frame.setSize(700,700);
        frame.setVisible(true);
    }


}
```

e) Cattle.java

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package wsn;

/**
 *
 * @author rajeev
 */
public class Cow {
  public  float bodyTemp,pulse,pedo;
  public Energy energy;

    public Cow()
    {
    bodyTemp=0.0f;
    pulse=0.0f;
    pedo=0.0f;
    energy=new Energy();


    }
    public float getbodyTemp()
    {
    return bodyTemp;
    }
public float getpulse()
    {
    return pulse;
    }
public float getpedo()
    {
    return pedo;
    }

public void setpedo(float a)
    {
    pedo=a;
    }
public void setbodyTemp(float a)
    {
    bodyTemp=a;
    }
public void setpulse(float a)
    {
    pulse=a;
    }
}
```

h). Cattle Prediction.java

```
  package wsn;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;


public class CattlePrediction
{
      public  Connection con;
      public Statement stmt;
      public ResultSet rs=null;
    public PreparedStatement pstmt=null;

    public void predictCattleData()
    {
      try
      {
             Class.forName("oracle.jdbc.driver.OracleDriver");


con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","Anshul","anshul");

            stmt=con.createStatement();
            String
tables[]={"CATTLESDATA","CATTLESDATA1","CATTLESDATA2","CATTLESDATA3","CATTLESDATA4"};
             Double temp,pulse,pedo;
         //    int n=stmt.executeUpdate("Delete * from CATTLEPREDICT");
            for(int i=1;i<=36;i++)
            {
                  temp=0.0;
                  pulse=0.0;
                  pedo=0.0;
             for(int j=0;j<5;j++)
             {
                    rs=stmt.executeQuery("Select * from "+tables[j]+" where CattleNo="+i+"");
                    while(rs.next())
                    {
                    temp+=rs.getDouble(2);
                    pulse+=rs.getDouble(3);
                    pedo+=rs.getDouble(4);
                    }
               }
                   temp=temp/5;
                   pulse=pulse/5;
                   pedo=pedo/5;
                 int          n=stmt.executeUpdate("Insert          into          CATTLEPREDICT
values("+i+","+temp+","+pulse+","+pedo+")");

              }
              con.close();
           }
        catch(Exception e)
        {
               //e.printStackTrace();
               //System.out.println(e+"hello");
        }
    }  }
```

i). CattleAnalysis.java

```
package wsn;

import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Dimension;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

import javax.swing.BorderFactory;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.border.Border;
import javax.swing.table.DefaultTableModel;

public class CattleAnalysis extends JFrame
{
        private JButton b1;
         //  private JPanel p1,p2;
           public  Connection con;
               public Statement stmt;
               public ResultSet rs=null;
          public CattlePrediction cattlepredict;
        public void showCattleAnalysis(Wsn wsn)
        {
               JFrame frame=new JFrame();
               frame.setLayout(new BorderLayout());

               frame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
```

```
            DefaultTableModel model=new DefaultTableModel();
            JTable table=new JTable(model);
            //table.setAutoResizeMode(JTable.AUTO_RESIZE_OFF);
            model.addColumn("Group No");
            model.addColumn("Cattle No");
          model.addColumn("Body Temp");
            model.addColumn("Pulse");
            model.addColumn("Pedo");
            model.addColumn("Analysis");
            model.addColumn("Action");
            table.getTableHeader().setDefaultRenderer(new SimpleHeaderRenderer());

            JScrollPane                                                    scrollpane=new
JScrollPane(table,JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,

JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);
            cattlepredict=new CattlePrediction();
            cattlepredict.predictCattleData();
            try
      {
            Class.forName("oracle.jdbc.driver.OracleDriver");


con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","Anshul","anshul");

           stmt=con.createStatement();
           rs=stmt.executeQuery("Select * from CATTLEPREDICT");
           int k=0;
           int grp,cattle;
                  while(rs.next())
                  {

                        if(k%4==0)
                        {
                              grp=(k/4)+1;
                              cattle=1;
                        }
                        else
                        {
                              grp=(k/4)+1;
                              cattle=(k%4)+1;
                        }
                        Double temp=rs.getDouble(2);
                        Double pulse=rs.getDouble(3);
                        Double pedo=rs.getDouble(4);
                        String status="";
                      String action="";

                        if(temp>42||pulse<14||pulse>35||pedo>55)
                        {
                              action="Urgent Action Required";
                        if(temp>42)
                        {
                              status+="Regular Fever ";
                        }
                        if(pulse<14)
                        {
                              status+="Pulse very slow ";
                        }
                        if(pulse>35)
                        {
                              status+="Pulse very high ";
                        }
```

```
                                if(pedo>55)
                                {
                                        status+="Urgent Rest Needed";
                                }
                                }
                                model.insertRow(k,                                        new
Object[]{grp,cattle,temp,pulse,pedo,status,action});
                                k++;
                        //      Object row[]={grp,cattle,temp,pulse,pedo};

                        }

                con.close();



        }
        catch(Exception e)
        {
                JOptionPane.showMessageDialog(null,"Cannot connect to Database");
        }

        //      p2=new JPanel(new FlowLayout(FlowLayout.CENTER));
                b1=new JButton("Back");
                 Border border=BorderFactory.createLineBorder(Color.black);
                 b1.setVerticalAlignment(JLabel.CENTER);
                  b1.setHorizontalAlignment(JLabel.CENTER);
                  b1.setBorder(border);
                  b1.setPreferredSize(new Dimension(50,50));
                  b1.addActionListener( new ActionListener( ) {
                      public void actionPerformed( ActionEvent e )
                      {
                         setVisible(false);

                         wsn.Gui();
                         dispose();

                      }
                    });
                //p2.add(b1);

           //frame.add(p1);
                //frame.add(p2);
                frame.add(scrollpane);
                frame.add(b1,BorderLayout.SOUTH);
                frame.setSize(700,700);
                frame.setVisible(true);
         // frame.setLocationRelativeTo( null );
        }
        public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub

        }
}
```