UNIVERSITY OF VAASA

FACULTY OF TECHNOLOGY

TELECOMMUNICATION ENGINEERING

Hafiz Muhammad Kamran Ul Haq

CLOUD COMPUTING: SERVER CONFIGURATION AND SOFTWARE IMPLEMENTATION FOR DATA COLLECTION WITH WIRELESS SENSOR NODES

Master´s thesis for the degree of Master of Science in Technology submitted for inspection, Vaasa, 6 May, 2013.

Supervisor Professor Mohammed Salem Elmusrati

Instructor M.Sc. (Tech) Tobias Glocker

ACKNOWLEDGEMENT

I would like to express my appreciation to the people who advised me and shared their knowledge during the whole period of the thesis. First of all I am thankful to Professor Mohammed Elmusrati and Tobias Glocker for the tremendous guidance in this thesis. My appreciation goes to the staff of Technobothnia for arranging the separate desk and the system. I am also thankful to Anas Ashfaq for sharing knowledge about website development. I want to thank all the people who always encouraged me and motivated me to continue to work hard. At last, I sincerely show gratitude to my instructor Tobias Glocker to benefit me in completing this thesis work.

TABLE OF CONTENTS **Page**

UNIVERSITY OF VAASA
Faculty of Technology
Author: Hafiz Muhammad Kamran Ul Haq
Topic of the Thesis: Cloud Computing: Server Configuration and Software Implementation for the Data Collection with Wireless Sensor Nodes
Supervisor: Professor Mohammed Salem Elmusrati
Instructor: Tobias Glocker
Degree: Master of Science in Technology
Degree Programme: Degree Programme in Information Technology
Major of Subject: Telecommunication Engineering
Year of Entering the University: 2011
Year of Completing the Thesis: 2013 **Pages:** 94

ABSTRACT

In the recent years, Cloud Computing has become very popular and an interesting subject in the field of science and technology. The research efforts in the Cloud Computing have led to a number of applications used for the convenience in daily life. Cloud Computing is not only providing solutions at the enterprise level but it is also suitable in organizing a centralized database which is accessible from every corner of the world. It is said that, 10 to 15 years later when all the enterprises have adopted the Cloud Computing, there will be no more perception for the data center in the company.

The aim of this Master's thesis "Cloud Computing: Server Configuration and Software Implementation for the Data Collection with Wireless Sensor Nodes" was to integrate the Wireless Sensor Network with Cloud Computing in a such a way that the data received from the Sensor node can be access able from anywhere in the world. To accomplish this task, a Wireless Sensor Network was deployed to measure the environmental conditions such as Temperature, Light and the Sensor's battery information and the measured values are sent to a web server from where the data can be accessed. The project also includes the software implementation to collect the sensor's measurements and a Graphical User Interface (GUI) application which reads the values from the sensor network and stores it to the database.

KEYWORDS: Cloud Computing, Wireless Sensor Network, GUI application, Sensor node.

1. INTRODUCTION

During the past several years, the enhancement in Wireless Sensor Networks and the development in Cloud Computing have a significant growth in the field of Information Technology. A Wireless Sensor Network emerges as a large number of sensor nodes which can organize themselves and operate autonomously. Nodes are very small in size, limited in energy source and limited in memory but equally capable of sensing, controlling, storing, transmitting and receiving. The information is collected with the sensor node and transmitted to a base station for further processing. According to Slijepcevic, Iyer & Panossian (2005) "[A] Wireless Sensor Network can be deployed in a more cost-efficient and robust manner than it is now."

On the other hand, Cloud Computing is relatively new and one of the most interesting area of Information Technology. Today, Cloud Computing offers the services which are completely suitable for enterprise architecture. Cloud Computing provides an architecture which is capable of implementing a number of services like external user, access control management, an interaction container, policy management and utility computing. Furthermore, Cloud Computing is an enhanced version of the Internet technology which provides the possibility of hardware hosting virtually in an external data center, platform services and application hosting. According to Jadeja & Modi (2012) "[Due] to the large scale proliferation of the Internet around the world, applications can now be delivered as services over the Internet. As a result, this reduces the overall cost."

The aim of the thesis was to integrate the Wireless Sensor Network and Cloud Computing. For this purpose, we need to deploy a small Wireless Sensor Network which will transmit the information to the Sink Node. After this step, the data will be further processed and uploaded to the web server. In **Figure 1**, the Access point is connected to Laptop or a computer which is capable of uploading the

information to the web site.

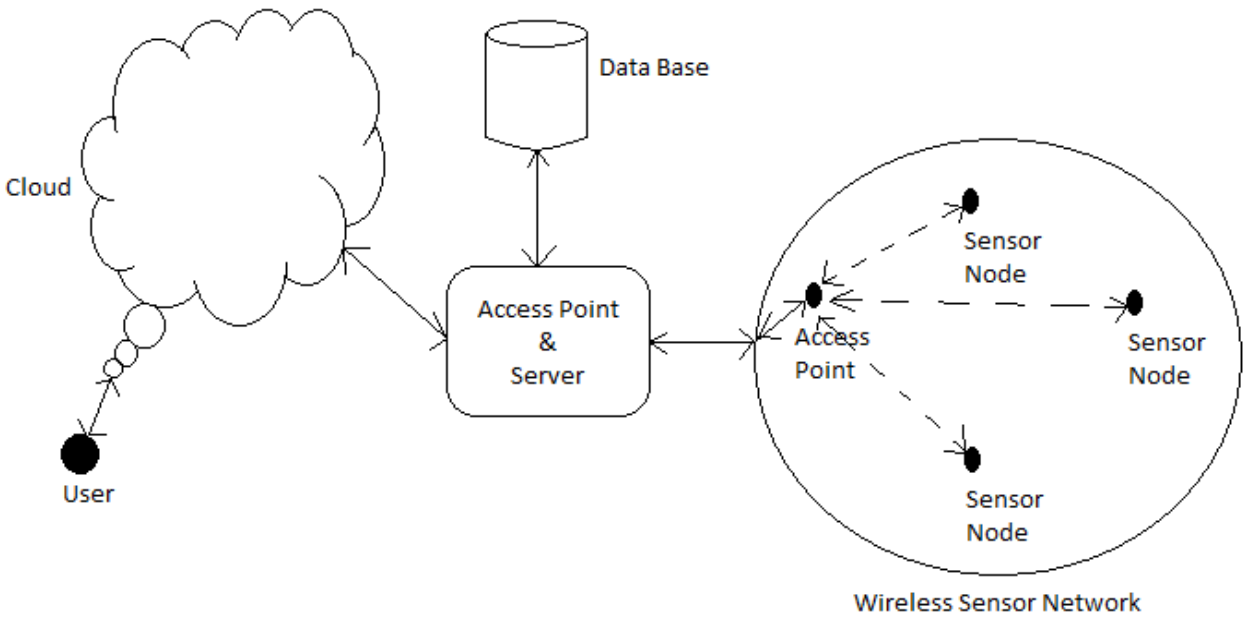

Figure 1. Integration of Wireless Sensor Network and Cloud Computing.

The thesis organized in six chapters. The first three chapters describe the theoretical concepts and explain the Wireless Sensor Network and Cloud Computing. The rest of the chapters include the implementation of the hardware and software. Furthermore, it is explained how the Wireless Sensor Network is deployed and how Cloud Computing is utilized. It also includes the server configuration, website development and graphical user interface. Moreover, there is also an experimental part of the thesis and the last chapter contains the conclusion and the future work.

2. CLOUD COMPUTING

Cloud Computing is a computing paradigm which provides on-demand network access to a computing resource (e. g. , network services, storage, servers and applications) and enables an ubiquitous, convenient and dynamic platform that can be released with minimal management effort or service provider interaction. (The U.S. NIST-800-145)

Figure 2. Cloud Computing (Linux Services Organization 2013).

This defines cloud computing as a utility in which you pay as much as you consume the services. **Figure 2** elaborates how devices are connected virtually to the cloud. It is similar with a telephone connection in a house where you pay as you use. Once a user is connected with the services offered by the cloud, they can consume as much, whenever and wherever they want and they will be billed for the resources they have used. (Winans & Brown 2009.)

Cloud Computing also offers a shared, scalable and elastic resource pool including computing power, elastic storage and software availability online to the users. Computing resources are meant to serve the multiple consumers using a multi-tenant model where resources are dynamically

assigned and reassigned according to the consumer demands. It means the users are able to run any software without fulfilling the requirements and configuration needed by the software and can store as much data as they want because of the elastic nature of the data center. All they need to do is to have a computer and an Internet connection. For example, in an enterprise, there are 25 employees and all of them have the different tasks to do including the requirement of different software. Now, the less efficient, time consuming and costly way is to establish their own data center, by buying all the latest machines along with the software and maintenance. On the other hand, they can buy the cloud services for the employees, which is an efficient and cost effective way to do the job. (Habib, Hauke, Ries & Muhlhauser 2012.)

2.1 Architecture of Cloud Computing

Cloud computing is emerging as a model to support "everything-as-a-service" (XaaS). According to Lenk, Klems, Nimis, Tai & Sandholm (2009) "[this] is a virtualized physical resource, architecture and business application platforms provided and consumed as services in the Cloud". A Cloud computing system can be divided into two sections: the front end and the back end. They interact with each other through the Internet or sometimes through a network. The side which is visible to the user/client is the front end whereas the system cloud is the back end. The back end is responsible to provide the cloud computing services like storage, software and various computing platforms. The client's computer gets the access to the cloud and uses the required applications. Administrating the system and client demands are done by a central server. It follows certain protocols and uses special software called the middleware. (Jadeja & Modi 2012). It allows network computers to communicate with each other.

Figure 3 provides the architecture of cloud computing and the important building blocks involve inside.

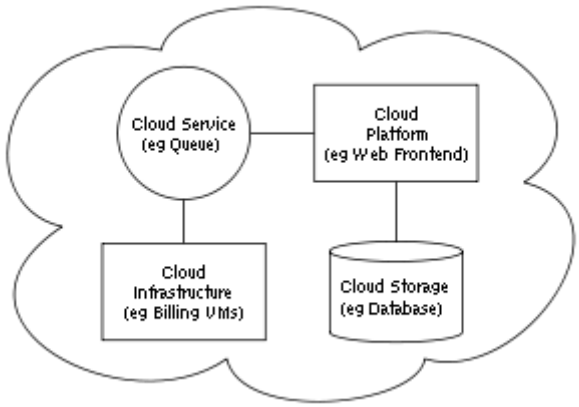

Figure 3. Architecture of Cloud Computing (Wikipedia 2013 a).

2.2 Cloud Computing Services

There are different types of services provided by cloud computing, but the most important three services are: Software as a Service, Infrastructure as a Service and Platform as a Service.

2.2.1 Software as a Service

Software as a Service (SaaS) is the model of software deployment in which a provider licenses an application to customers for the use as a service on demand. (Zhou, Zhang, Zeng & Qian 2010). It gives subscribed users access to software or services that reside in the cloud and not on the user's device. The consumer of software as a service application only needs a basic application (web browser) to run the services provided by the service provider. It efficiently reduces the requirement

of maintaining and installing the hardware and software at the end user side and provide the centralized control and deployment of the software. The popular SaaS applications are Hotmail, Gmail and Google apps. (Gibson, Rondeau, Eveleigh & Tan 2012.)

Software as a Service has numerous advantages as it is a cost effective solution and has remarkable effects when making a budget for an organization. Microsoft Corporation emphasizes on deploying SaaS applications which has low initial investment cost on software, hardware and staff. Another study performed by (Hurwtiz & Associates) summarized that "SaaS solutions offered 64% savings over 4 years for a comparable on premise solution". (Brodkin 2010). **Figure 4** explains how the services provided to the users through the Cloud.

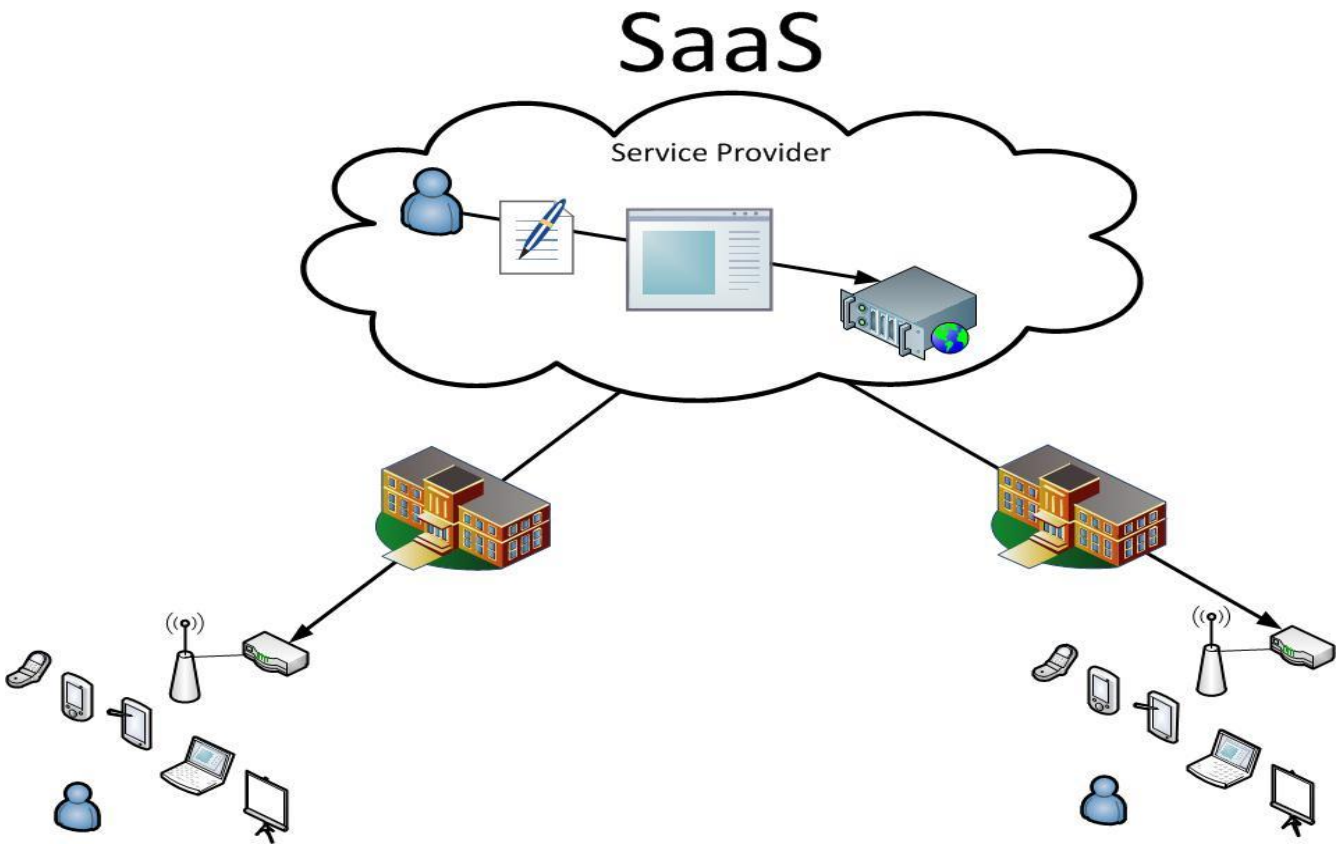

Figure 4. Software as a Service (SaaS) (Brad 2011).

SaaS provides benefits to an organization when it comes to the security of data. SaaS based application provide the flexibility to store the information in a centralized database and also eliminate the need to carry sensitive information all the time. It would be impossible to figure out all the possible locations of data without having a centralized location for an enterprise (which includes heavy costs and liability). Furthermore, a centralized SaaS provider can be much more efficient when it comes to incident response. Further security benefits include shared security testing costs, deploy secure logging and efficient systems. (Brodkin 2010).

Accessing data anywhere is convenient, but it is risky when it is insecure. Many companies provide Cloud services, such as Media Temple, AT&T, and Grid Player and so on. All of them have similar security properties. For example, a layered security model provides multi-level protection of all information, data and physical assets including data center environments. This model includes: network security, physical security, firewall management, virtual guest security, virus and patch management, and access controls and data security performed by encryption.

2.2.2 Infrastructure as a Service

Infrastructure can be defined as a set of hardware components including, capacity, storage, network, memory and so on (Zhou et al. 2010). These components are delivered as a service over the cloud to the end user and are charged in terms of usage. This service can also include the use of servers and virtualization to enable the utility for the end user (Gibson et al. 2012). IaaS provides web based services to create, control and manage the virtual machines and storage. These services will produce bills (after a period of time) for the users according to the agreement. It will eliminate the responsibility of managing the physical and virtualized infrastructure while still running the software on the virtual machine. **Figure 5** shows the Infrastructure as a Service provided to the

users through the Cloud.

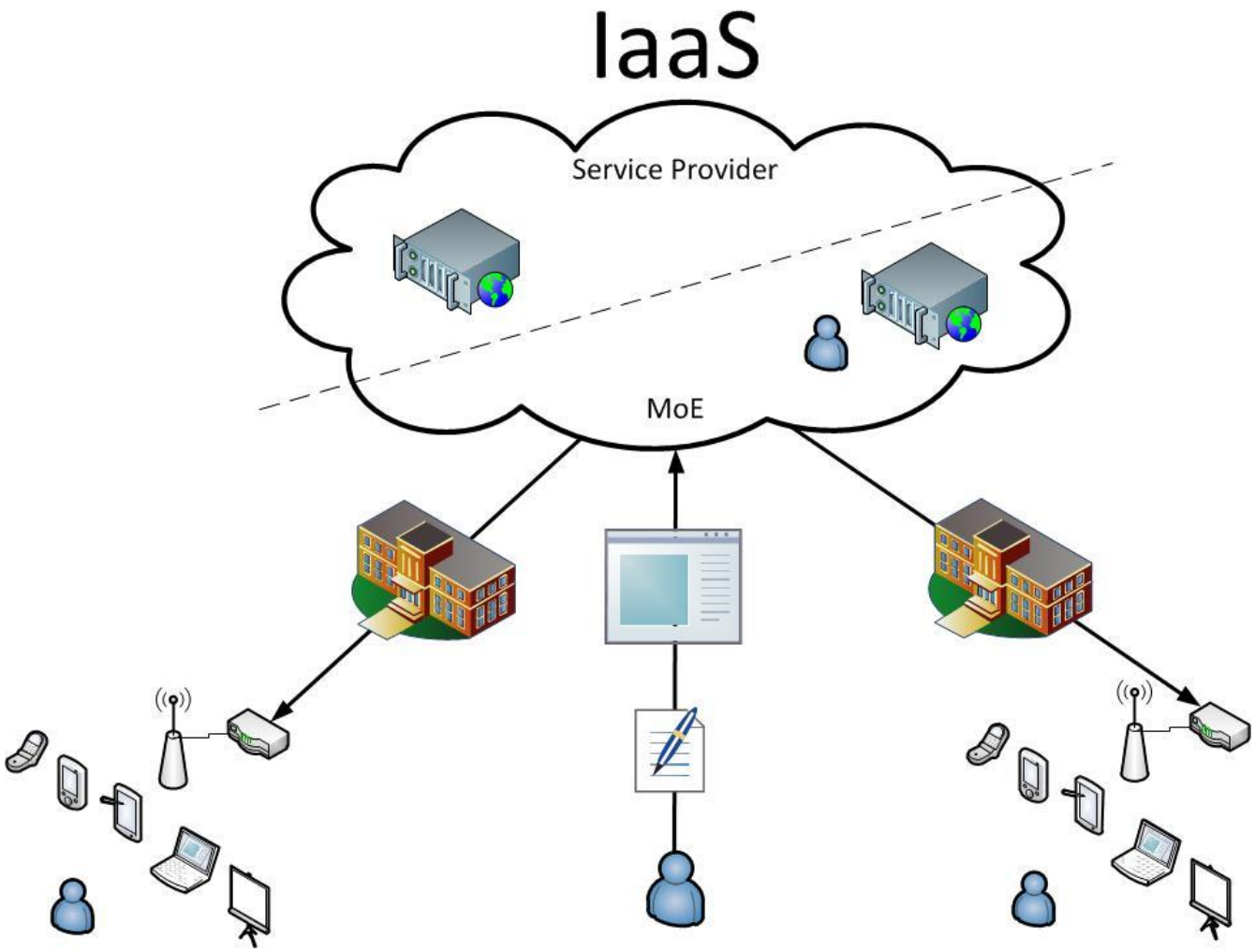

Figure 5. Infrastructure as a Service (IaaS) (Brad 2011).

One of the challenges faced by the management is the decision to operate either self-managed or subscribed to a hosted service. A self-hosting the infrastructure may be more cost effective in terms of bandwidth and computed loads, while the other solution seems to be fine for light and intermittent use. There are still some factors that need to be considered, such as security policies, location of the data administration when making the decision for the hosted services.

Security is an important concern especially in a shared environment. An enterprise might be hosting many other enterprises work load and data which may expose all parties to a risk of security related incidents. For example, using virtualization the hypervisor has privileged access to the physical resources. A system administrator may not have access to the guest operating system running within a client virtual machine (Gibson et al. 2012.)

Assigning ideas carefully to personnel, applying security principles of least privilege would be an overcome to the security challenge. In the context of communication, encryption of data has to be secured within the guest operating system and beneficial to protect the virtualization layer.

2.2.3 Platform as a Service

Platform as a Service (PaaS) delivers an operating system and associated services. To the famous service providers belong Salesforce, Google and Microsoft. These service providers offer access to the API, which allows users to develop and customize applications without the installation of the development environment. PaaS has identical benefits as IaaS and SaaS such as utility computing, resource allocation, virtualization and cost reduction. A preconfigured environment reduces the setup time and administration time. (Zhou et al. 2010.)

The development is completed in back end using the tools to build the applications and services provided by the cloud platform. After this, the application can be delivered to the end user through the cloud. Compatibility is one of the most important challenges for the utilization of the PaaS and there is no way to choose the right service provider (Gibson et al. 2012). **Figure 6** shows an image of PaaS which is provided to the users through the Cloud.

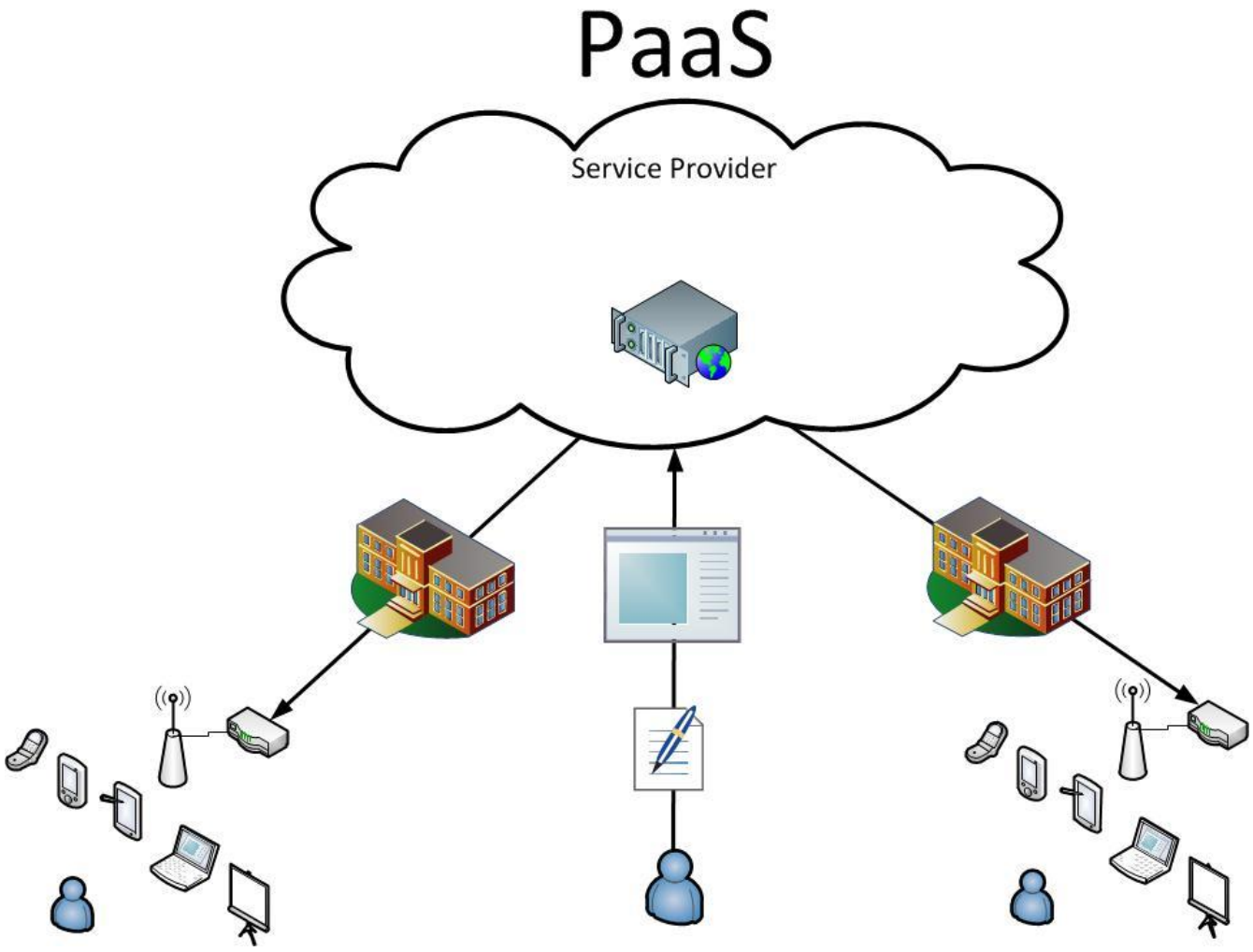

Figure 6. Platform as a Service (PaaS) (Brad 2011).

The Cloud platform can be divided into two categories as full or partial PaaS. Full PaaS provides a complete platform application for the user. Partial PaaS provides the tools of the cloud platform to the user as a service. In Partial PaaS, the applications and solutions on the client's computer must be installed. Security plays essential role in PaaS. If the services of PaaS are shared in the public cloud, it will limit the capabilities of the data security as compare to an enterprise cloud. The most common examples of PaaS are Azure Service Platform, Google App Engine, Force.com and Engine Yard.

2.3 Service Reliability and Availability

For any complex system, failures are inevitable. The system error can subject to a failure of services at the user end. A failure can occur in the hardware or software. In of all these circumstances, the user will not get the service by the service provider (Bauer & Adams 2012: 26-60).
In the following sections, the different sources of errors will be discussed.

2.3.1 Errors and Failures

The defects in the hardware or the software residuals can produce occasional errors. These errors eventually make an impact on the system operations and causing failures. If the system's initial error has not overcome, it results in one of the cascaded errors. These failures can be the corruption of an IP packet in a transmission or it can be a software failure which crashes the system process. The primary characteristic of the service impact of a failure is the duration of the service disruption. The problem can be mitigated by a very simple method such as retransmission of the packet or service and the impact of this would be a little bit long latency of the service. On the other hand, longer service disruption will make it visible to the user. For example, if some packets are lost during a real time service then the user will experience jitters in a video or a degraded sound in a voice call.

2.3.2 Eight-Ingredient Framework

The eight-ingredient framework or 8i, is a model used for methodically expressing all the potential of the system vulnerability, developed by Bell Labs (Rauscher 2006). This model enlightens us about: software, hardware, power, environment, payload, network, human and policy.

Each of these components plays an important role and has vulnerabilities which lead to errors and failures.

1-Hardware:

Systems are made by the electronic components. Hardware can lead to physical failure. In this way, suppliers are expected to provide the system with low failure rates, which depends on the manufacturing quality. Hardware should be designed in a way to provide visibility to the software so that in any consequence, if a hardware failure occurs, the software would be able to notify the failure and activate an automatic recovery procedure which restores the system to full operational status.

2-Software:

The software is responsible to provide an application layer to the user. Software can be prone to programming errors and the architecture design which can cause the system to abnormal behavior.

3-Power:

Power also plays an important role for any system. Electronic systems require appropriate AC or DC power and stabilization from over voltage, open circuit or short circuit.

4-Environment:

Hardware system needs suitable environment conditions to perform its job correctly. The critical environment conditions such as temperature, humidity, air density and so on should be well suited for the hardware. The system should be installed in an organized manner in order to avoid any problem or hazard.

5-Network:

Systems are supposed to be fit into a network when IP packets are the mode of communication with other systems. There can be number of devices involved for communication from one system to another. These devices include: optical fiber, router, Ethernet cables and switches. Eventually, any of these devices can lead to a failure which will disrupt the communication of the network.

6-Payload:

The interaction of the system with the user or other systems is done through the IP network infrastructure. Some of the elements involved in this interaction can be of different types. It is necessary during communication that all the elements should be separated from each other.

7-Human:

The routine and the emergency maintenance of the systems are done by the humans. There is a risk of damaging the system to the following: Human machine interface (HMI) is poorly designed, unavailability of the human, lack of training of a human and mistakes done under pressure.

8-Policy:

Operational policies are required in order to successfully interact among all the elements with the user and other systems. These policies often include industry standards, compliance strategies, maintenance and repair strategies and service level agreement (SLA).

2.3.3 Service Availability

When a failure occurs during any event for more than a few seconds, it will not only stop the user service request but also fails the retries for the same failed request. This will impact on the failure of a complete session or event and if the first retry fails because of the continuation of the service failure then it is considered as service outage. If this event goes on for a long time, the user will abandon the request for the considered service. In this way, we call it unavailability of service.

The service availability can be determined by simply dividing the service Uptime by the sum of service Uptime time and the downtime. Both of these values can be predicted by Markov availability Model or it can be the actual measured value.

$$\text{Availability} = t_u / (t_u + t_d) \quad (1)$$

Where,

t_u is the Uptime.

t_d is the Downtime.

Complex systems often experience the outage or failure of service which can either affect the user services or the other outages like redundancy and other factors. An event that causes a failure to the user services will be called service outages (Bauer & Adams 2012).

Every system whether it is based in an enterprise or if it is a Personal computer require maintenance after a certain period of time. A system which is meant to perform the IT jobs may require occasional maintenance such as upgrading the firmware, hardware, updating the software and

changing the position of the system. In order to make all these events possible, there should be a preventive plan or backup so that the services to the user would be uninterrupted.

An enterprise plans its update of software or maintaining windows during the night so the outage impact will be minimal to the user. It is a difficult job for the engineers to finish their job in a minimum time during the shutdown because they have to compensate the process execution. During the period of maintenance, the availability of service becomes possible due to the diversion of the events to the other system. After the maintenance or upgrade, the system needs to restart.

2.3.4 Service Reliability

Reliability is often considered as the superset of the service availability and some other factors. As Microsoft stated (2012) "[the] reliability ensures that service capacity, service availability, service continuity, data integrity and confidentiality are aligned to the business needs in a cost effective manner."

Service reliability can be measured easily, the successful response divided by the total requests. It means how many times a user service request has been successful and lead to an expected result. The more the service requests are successful, the better the service reliability.

$$\text{Reliability} = R_s \ / \ N_r \qquad (2)$$

Where,

R_s is the successful response.

N_r is the number of requests.

The transaction can lead to a defective service and cannot provide requested service because most of the application protocols provide return codes which might be used to identify the non-successful request to a failure. A failed service transaction can happen due to the following reasons: the server found an unexpected condition which prevented to succeed the request, the server is temporarily unavailable because of maintenance or overloading, request time out or the latency is greater than computed. (Bauer & Adams 2012.)

2.3.5 Delay (Latency)

The network based services allow communication transactions from the users. For example, for a telephone operator, initiating a call would be a response to a request by the user. Getting access to your inbox would be a response to the request for it, web applications return web pages in response to the HTTP GET request. In other words, latency is related to the quality of experience at the user end.

The latency between the times an application receives a request depends on Bandwidth, Caching, Request Queuing, and Variation in request arrival rates and network congestion (Bauer & Adams 2012).

Bandwidth:

Network bandwidth is an important factor for the service latency. The insufficient bandwidth from the user to the cloud or the cloud to the user will cause a delay.

Caching:

The response to any request done through cached memory which is much faster than that of reading the storage disk.

Queuing:

In a busy hour of the day, request queuing is very effective preventing the network congestion and each request treated one by one in an organized way. Logically a buffer implements the request Queue.

Request arrival rate:

Requests randomly occur on the network. The delay between the arrival of requests depends on the load of the system. A congested network will have a larger Queue and will have a higher delay for the request response.

$$\text{Delay} = (\text{time})\text{Arrival of request } (R_a) - \text{load factor } (L_f) \quad (3)$$

Where,

Load factor (L_f) = time required to execute one request – number of request.

2.3.6 Real Time Services

Real Time service requests are different from other requests. This request produces a massive transportation of data in Kilo Bits per Second (KBPS). For example, a video call, a voice call or an on line streaming. The difference of the result leads to a different reliability issue, service quality and availability risks.

Real Time services like video and audio have two fundamental components: Session control and user data. Creating a call, processing and terminating it, is the responsibility of session control where the session initiation protocol is used to control voice and video data. These protocols are also responsible for digital encoding and decoding of the signals.

Real time communication data is inherently uniform. It assures that the audio/video contents that were encoded by transmitter is transported correctly through the network and has been decoded at the receiver side in order to have a perfect communication. (Bauer & Adams 2012.)

Quality of Real Time services depend on the packet loss. The jitter is an important factor for the quality of the streaming services depending on the arrival rate of the data packets. It can be overcome by re arranging the sequence of the late packets with the current one.

2.4 Cloud capacity, elasticity and planning

One of the great feature of the cloud computing is elasticity which makes it different than the rest of the ordinary network models and from redundancy. Basically redundancy is designed to provide resources to reduce failures of the service capacity and it is expected to overcome the impact of load within the duration of seconds or minutes. The elasticity feature is designed to increase or decrease the capacity factor depending on the offered load which can be altered in hours, instead of weeks or months.

The system overload situation can be well understood by an example of the offered load. Within five minutes of a natural calamity, the system can be overloaded because the users are attempting to call their family or friends, which produced congestion on the system. If the service is not provided

sufficiently, the service performance will be degraded and the reliability and availability will be damaged.

2.4.1 Capacity Planning

Server reconfiguration like adding more RAM or a new Processor to the system requires shutting down for a while. The services offered from this server are supposed to be compensated to some other server. Upgrading the server is an important job while having significant benefits:

- Migration of the traffic towards the server being reconfigured
- The system should turn on once again when the configuration is done
- Restore the application, operating system used on that server
- Restarting the application
- Migrating back to the expanded system carefully so that impact is minimal to the user

The upgrade and reconfiguration seem to be a risky job. It involves humans and they may cause failures. It is an expensive and time consuming job that is the reason why most of the enterprises buy the new servers in order to avoid the time consumption on reconfiguration which also involves the risk of the failures.

Elasticity is an essential feature of the cloud computing which enables the additional resource to support the load of applications. The service license ensures the users that they will be charged only for the resources they have used. **Figure 7** illustrates how to make a balanced capacity planning.

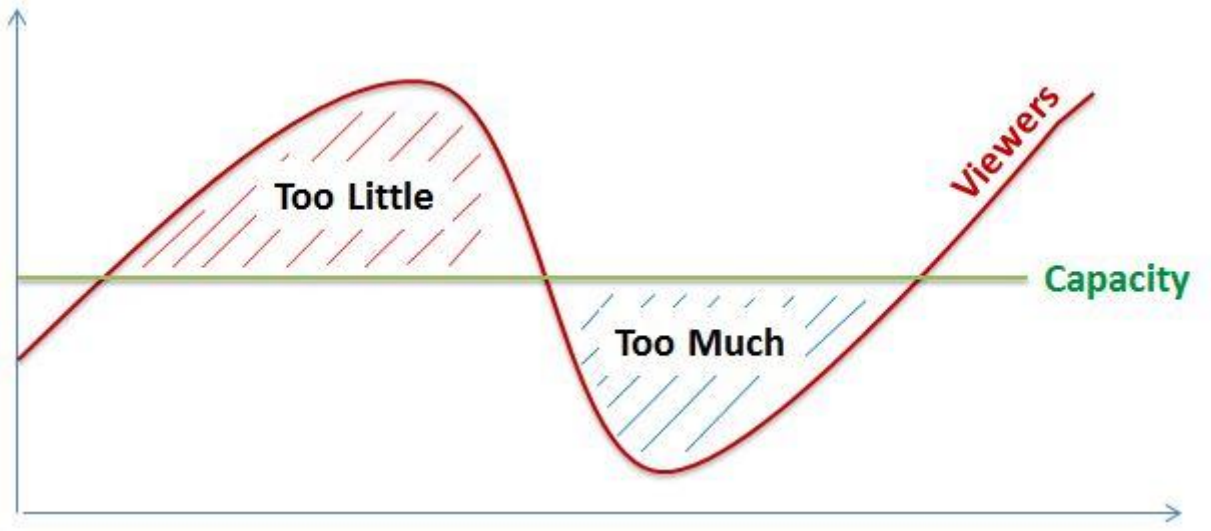

Figure 7. Capacity Planning (Nothern 2011).

Capacity growth has different types:

- Vertical growth: Instances of the virtual machine supposed to allocate more memory, network and CPU.
- Horizontal growth: More virtual machines can be spawned to meet the requirement of the offered load.
- Outgrowth: Instances of the application independently run in different data centers, which is the typical characteristic of the cloud.

2.5 Security

Some of the enterprises are reluctant to adopt cloud computing because it brings a lot of risk for the confidential data of the customers. Trade and personal information can be fallen into wrong hands. Security of the sensitive information and data needs some serious investment for the storage and control over the cloud. In cloud computing, the data is supposed to be stored in multiple locations geographically distributed, so that authorized persons can only access the information (Tianfield 2012).

Cloud computing provides the flexibility of integration with a number of applications and different platforms. This characteristic makes it a security risk. A cloud application should be capable of communicating with other applications without any security risk. There must be a firewall installed to prevent the system from any attack and at the same time the registered users should be able to access their data easily.

2.5.1 Ensuring Information Security

- One method to ensure the information security is to use an encryption method and to use a very unique key to decrypt. Many data centers use encryption to secure the data which prevents the access of the unauthorized user.
- Determining which Internet application is trust worthy and how they make profits.
- Some online activities may be difficult to protect even if the file is encrypted. In this way the user needs to carefully consider the risk of leakage of information to the marketers.
- Privacy policy is needed so that the data can be shared according to the service level agreement (Jia 2011).

2.5.2 Characteristics of the cloud for security

The cloud service providers use the multi-tenancy in which multiple users share the same resources, infrastructure, storage and application. Multi-tenancy allows cloud providers to manage the resource utilization efficiently by the virtualized infrastructure. Cloud computing provides multi-tenancy, so that the users experience flexibility, reliability and availability through on-demand provisioning. It also ensures the partition on the virtual environment for the multiple requests of services on the same physical server and prevents from any possibility of leakage (Tianfield 2012).
The cloud computing model includes multiple stakeholders: cloud provider, service provider and customer. Each of them has their own security management and expectations which includes the following issues:

- Each of the stakeholders are responsible for their own security management processes, expected risk and impacts and how to prevent from them
- Security properties should be mentioned on the service level agreement on which the service providers and customers agreed
- The security requirements can be different among tenants in multi-tenancy environment. These requirements are setup according to the customer wishes in SLA

Another issue of the security is the third party issue; it means the owner of the data has less control on the data processing or the partial control. The governance also needs to be considered because the customers will give up their own infrastructure and control.

3. WIRELESS SENSOR NETWORK

The role of a Wireless sensor network (WSN) in technology is huge, supporting different applications in the areas of military applications, industrial automation and health care. A WSN provides architecture with low cost, low power, sensing abilities on distributed nodes and network design. Small sized hardware components in WSN can be deployed easily and are robust with high efficiency.

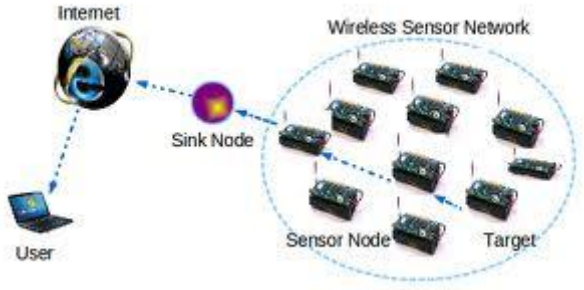

Figure 8. General architecture of WSN (Virtual-lab 2012).

The sensor network based applications are not new, but the technology along with the wireless characteristic has significant importance (Slijepcevic, Iyer & Panossian 2005). The aim of using the wireless sensor network in most of the latest research works is due to its ability to operate remotely without a particular infrastructure. These sensors are capable of processing and executing data such as temperature and high resolution images, which lead to the deployment of a wireless sensor network in most of the applications. **Figure 8** explains the basic architecture of WSN.

The infrastructure of a WSN consists of distributed, task oriented, independent, dynamic topology, and low power wireless nodes. The wireless nodes can communicate with each other by making a dynamic wireless link. WSNs consist of thousands of nodes which are distributed geographically, each of them collects the information from the environment and processes it and then sends it to the central point or the access point (not in the case of ad hoc network) to provide results. (Jangra, Swati, Richa & Priyanka 2010.)

The WSN nodes are small sized with limited memory, processing power and energy source. Every sensor node consists of the sensing capability (to collect the data from environment), microcontroller (for the processing), ADC (to convert the mechanical data into digital format), memory (to store the information), transceiver (for data communication), power unit (to supply power to the nodes). The information is transmitted from sensor nodes to the access point or the base station for further processing. This communication can be of two types: Single hop and Multi hop depending on the area covered by the network. There are three types of WSN: continuous, on-demand and event-driven. In continuous monitoring, the nodes sense the data continuously and send it to the base station periodically, depending on the specified time duration. In on-demand, the nodes sense the data and store it in the memory. In this case the data will only be sent to the access point on request. In event-driven, the nodes only transmit the data when an event occurs. (Baghyalakshmi, Ebenezer & Satyamurty 2011.)

3.1 Architecture of Wireless Sensor Network

WSN is classified into two types: flat and hierarchical. In flat type architecture, each sensor node transmits the data to the neighboring nodes. This neighbor node will forward the data to the access point. In this case, all the nodes have equal priority. In hierarchical architecture, one node assigned

as a cluster head. Every cluster head will receive the information from all the nodes and then transmit it to the base station. There can be different architectures in the WSN, depending on the specific applications and requirements.

For example, **Figure 9** gives architecture of a WSN. The sensor nodes are inside their consecutive clusters. For one cluster, there is one node which acts as an access point (Cluster Head) that forwards and receives the data of all sensors nodes (Slave node) inside the cluster. The nodes that are present at the intersection point of the two clusters provide the communication between these clusters is the Cluster Heads. In this way, the WSN is distributed in multiple clusters which reduce the power consumption and which are efficient for the data communication.

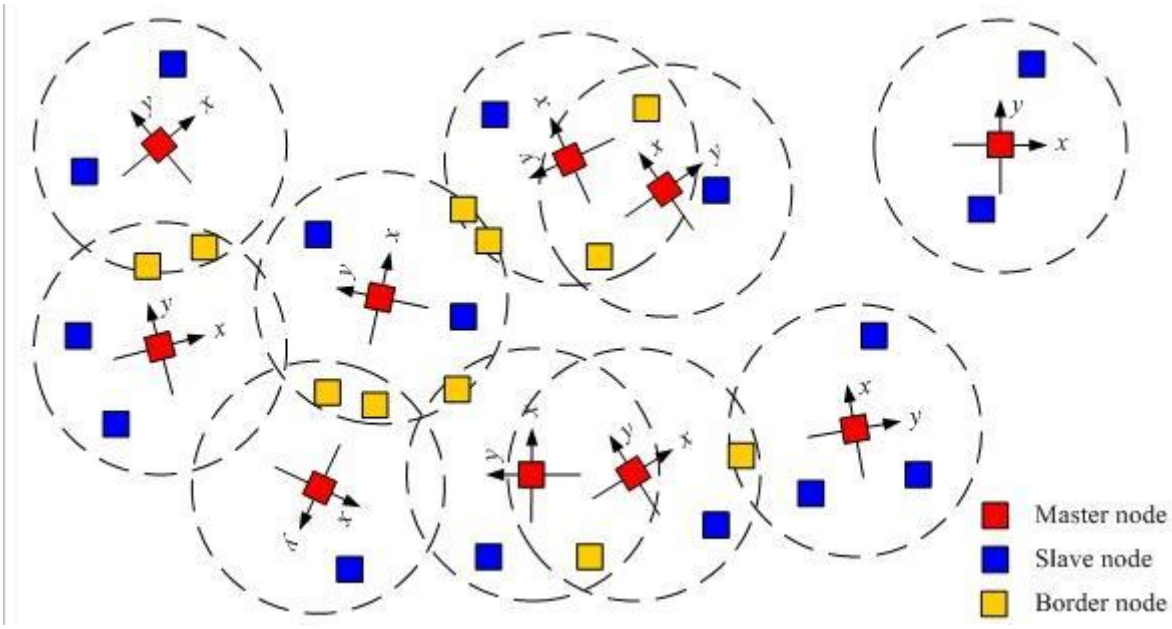

Figure 9. Example of the Architecture of WSN (WSN-advanced-computer-science 2010).

3.2 Characteristics of Sensor Node

- Energy efficiency: Resources should be utilized properly, it would be impossible to recharge them again.
- Low cost: A sensor network can consist of hundreds of sensors, so a low cost solution for each sensor node must be provided.
- Distributed Sensing: Robustness is very important for the WSN. It can be achieved by distributing the sensors in the network so that they can collect and store the required data.
- Wireless characteristics: Sensor nodes are considered to be wireless in nature. In that way, they can be deployed easily without the need of the specific infrastructure or back backbone network.
- Multi hop: When a large sensor network is deployed, multi hop nature becomes an essential part for the communication of the nodes with the access point. Each sensor node is not capable of communicating to the base station directly because of the limited resources. So they need to send the data to the neighboring nodes and which then forwarded to the base station. A large number of nodes can be involved in this communication; it is an energy efficient way of communication.
- Distributed processing: The sensor nodes collect the data and on some level process it to make information (Jangra et al. 2010).
- Node type: On the basis of capability and range of the sensor node, there are two groups of sensor nodes. Homogeneous node and Heterogeneous node. In Homogeneous group, the sensor nodes have the same sensing capability and are identical characteristics, while Heterogeneous group, the sensor nodes have same capability but have different characteristics with each other.

Figure 10 shows the Multi Hop characteristic of WSN.

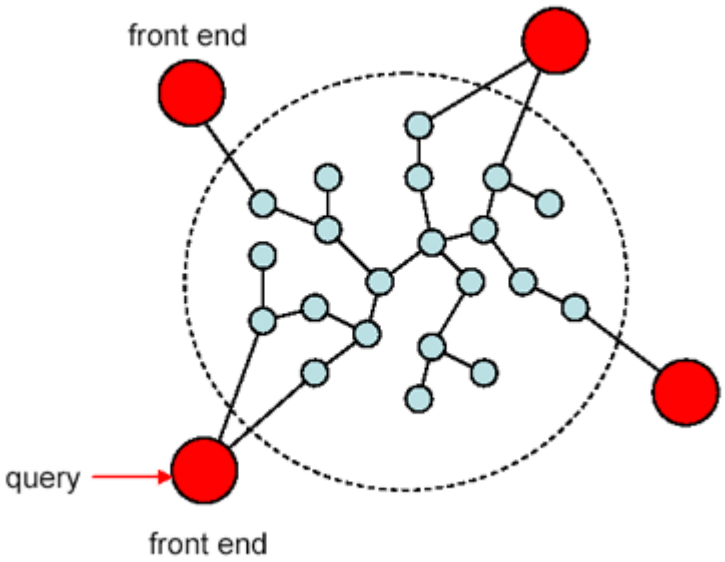

Figure 10. Example of multi hop network (Liu & Gedik).

3.3 Power/ Energy Management

A distributed sensor network is emerged as the solution of different applications for the data collection in a wide range. The most difficult challenge faced by the designers is the reduction of power consumption. Power management of the sensor network focuses on the scalability, reliability of data, obtaining a desired result and energy consumption. The desired architecture of the WSN appeals an efficient software approach which provides energy quality trade-off as well as the suitable energy consumption of the hardware. (Hac 2004: 30-60.)

The energy efficient design points toward micro and Nano sensor applications. Energy consumption at hardware level, effects due to low duty cycle (DC) operation of the sensor node. This design can be adaptive in an active workload condition. At software level, energy consumption can be minimized by adapting an efficient algorithm (low power algorithms).

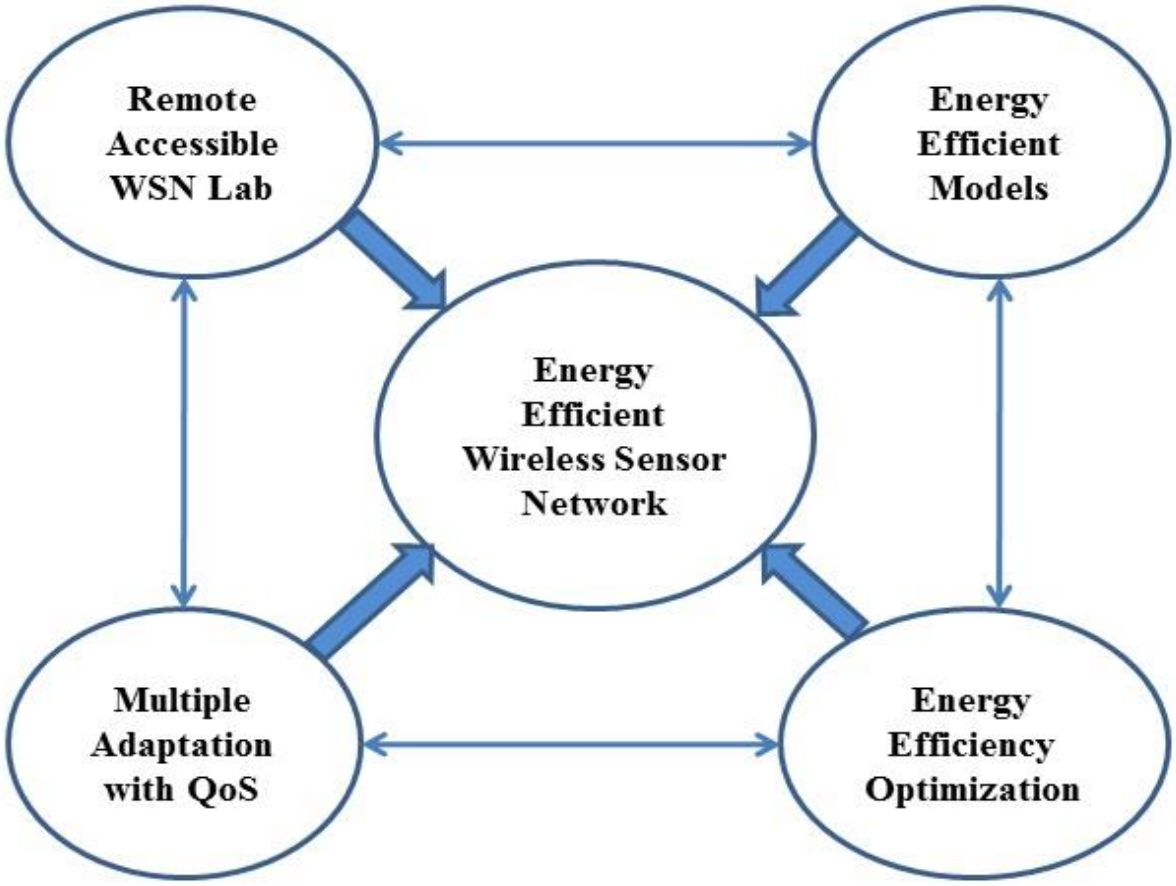

Figure 11. Example of Energy efficient WSN (TSU 2008).

The energy efficient algorithm is highly desirable for a WSN. Although energy scalability increases the computational complexity. Distributed sensor networks are a good choice for providing scalable and robustness environmental monitoring. The power management technique is used in most of the distributed WSNs. It provides the efficiency to the embedded operating system by making a

transition in sleep mode, which are based on the analysis of the events (Hac 2004). **Figure 11** enlightens how to achieve an energy efficient WSN.

Energy efficient system designs are necessary for WSN. Based on the distributed number of nodes which can increase the quality, scalability and robustness, the network should adjust itself in different conditions and provide desired results without any quality compromise.

3.4 Energy Efficient Communication

Distributed sensor networks collect the information from the environment without maintenance and fault. Micro and Nano sensor networks provide good quality characteristic such as high node density, low data rate and reduced energy consumptions. Micro and Nano sensor can provide the high quality results for a long duration equivalent from 5 to 10 years depending on the conditions with just a normal cell battery.

An effective micro and Nano sensors stack have the energy efficient protocols and can communication with the different applications. They can also investigate the network protocols, MAC layer and Radio frequency.

In order to achieve energy efficient communication, the sensor nodes adopt the approach to multi hop routing protocols which reduces the path loss occurred by the medium. There are two types of routing algorithms: Source routing and distance vector routing. In source routing protocol, the information is transmitted to the base station by the intermediate node (next hop) while in distance vector routing, each node maintains the table for its next hop to find the shortest way towards the base station.

Traditional way fails to minimize energy consumption, if we consider the protocol overhead. Although the MAC layer have been adopted universally to reduce the energy consumption. IEEE 802.11b requires high power consumption which is not suitable for WSN. In a multi hop routing, the packet overheads are compensated by the shorter radio frequency transmission. Multi hop routing is beneficial at the transmitter, if we consider the transmission distance large.

3.5 Sensor Node

A sensor node refers to a single element of a WSN shown in **Figure 12**. The major concern of the designers bounds them with the efficiency of each node. The sensor nodes can be deployed in a location which is not specified and have no particular infrastructure (Ad hoc) which makes it a complex task to pursue. In order to utilize the sensor nodes in an effective way, the area and the distribution should be specified. (Slijepcevic et al. 2005.)

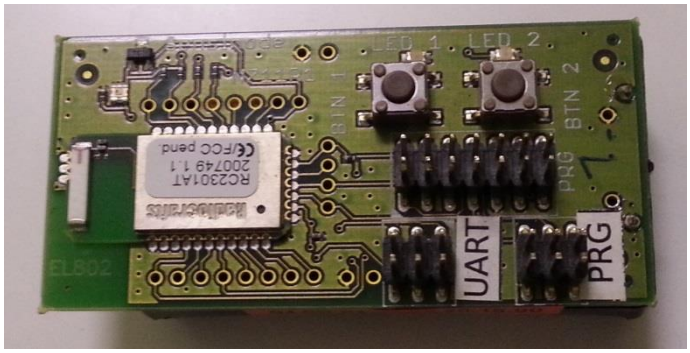

Figure 12. Sensor node.

The architecture of sensor node (see **Figure 13**) consists of a sensing unit, MCU, communication device (transceiver) and a power supply. A sensing unit comprises on one or multiple sensors depending on the requirements (temperature sensor, humidity sensor etc.). The microcontroller unit (MCU) basically executes the information provided by the sensor or stores it in the memory. The transceivers are responsible for the data communication of the sensors. Power supply is necessary to operate all the mentioned functionalities.

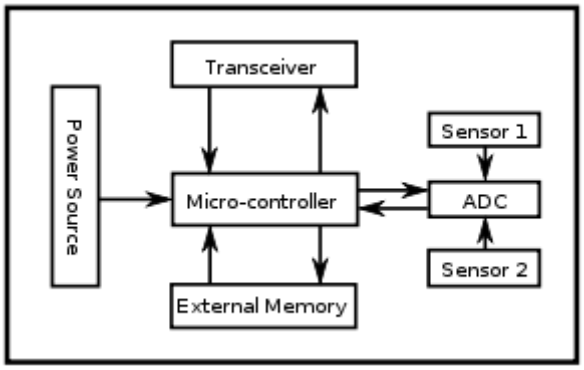

Figure 13. Block diagram of Architecture of sensor node (Wikipedia 2013 b).

3.5.1 Sensor Network Layers

Physical layer:

Sensors use RF transceivers for the communication. RF transceivers consume low power and low frequency. But in fact, to achieve such conditions, a large antenna is required. A large antenna disturbs the dimensions of the sensor nodes. It is also possible to use IR sensors but the problem is, it needs line of sight to communicate with each other.

Distributed sensor networks can provide the solution to this problem by using a lower transmission

range and by allowing multi hop communication. But still, there will be a trade-off over communication overheads.

MAC protocol:

Sensor nodes can communicate with each other with a low data rate. This will enhance the delay of the communication which is often a secondary priority in WSN in most of the applications. Mac protocols are implemented as TDMA frame. Power consumption is reduced by setting the communication on and off. It means, the power will turn on when it is transmitting or receiving a data packet. This approach requires synchronization.

Network layer:

This layer is responsible for initiating the communication in the network. The network layer ensures the secure and reliable transmission from the sender to the receiver. This communication involves addressing of the sender and the receiver and the routing information in the network.

Application Layer:

The application layer makes sure that the data sent to the user is in a correct format. It includes the interpretation of a machine language to the human understandable language.

3.5.2 IEEE 802.15.4 and 6LoWPAN

IEEE 802.15.4 is a standard for low power and low data rate wireless communication between small devices. 802.15.4 is not similar with ZigBee. 802.15.4 is a MAC layer protocol while ZigBee is network layer protocol.

Specification:

- 2.4 GHz ISM band (Q-QPSK at 250 kb/s). It can operate on several bands.
- 2.4 GHz output power (20 dBm)
- Data Rate up to 250 kb/s which depend on band and its mode. It can achieve the speed up to 625kb/s in Turbo mode.

The 802.15.4 network enables a beacon. The coordinator PAN a transmit beacon frame to synchronize. It provides access to the slotted channel. A beacon enabled network allows devices to consume less power which is achieved by the receiver switching off the parts of the super frame.

Meshing is the ability to transmit the data from sender to receiver using multi hop. It is not actually part of 802.15.4 standard but it operates from the network layer.

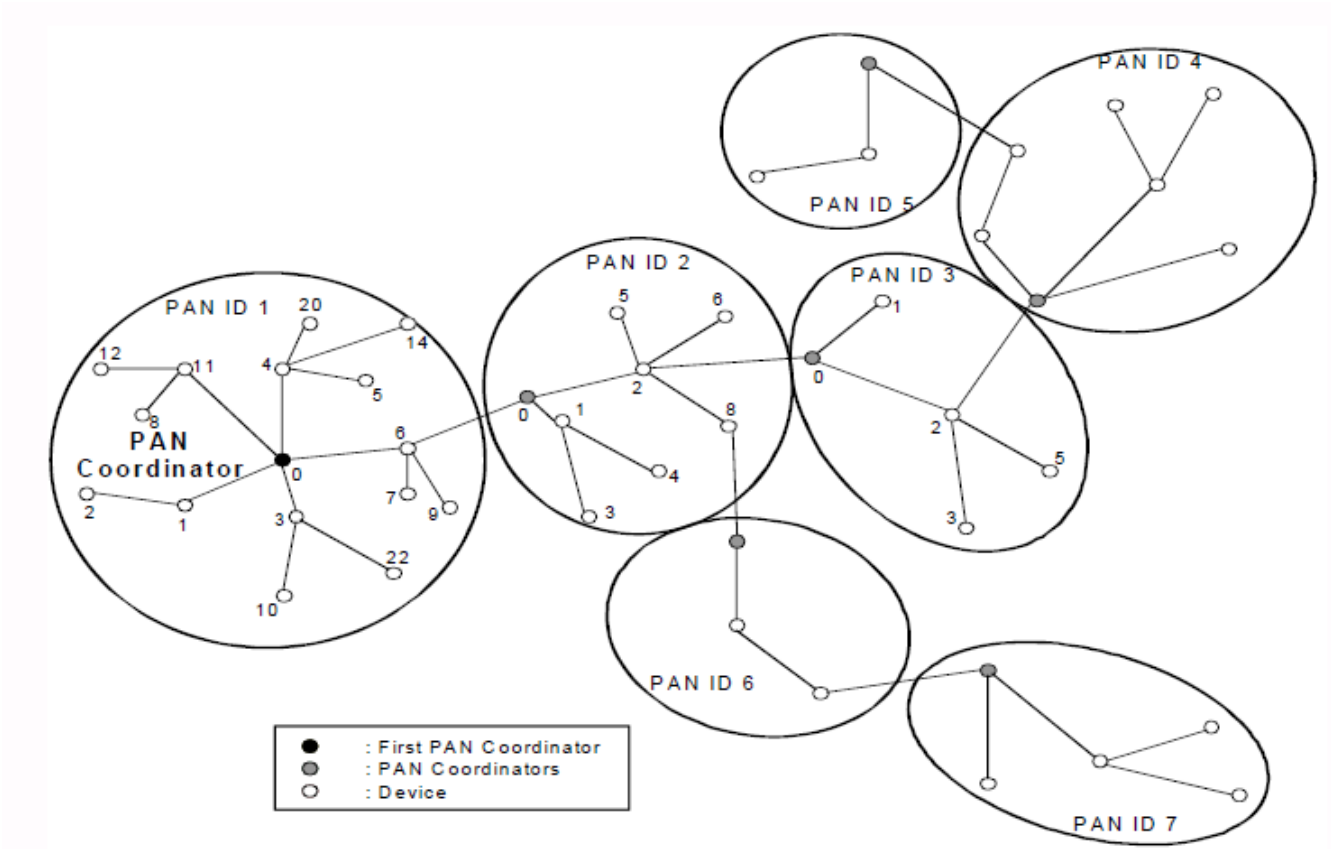

Figure 14. Meshing in PAN (Ott 2012).

Figure 14 illustrates that each PAN has a PAN coordinator and a Full-Function device (FFD) which is responsible to process the requests of joining or leaving the network. It also assigns the addresses to the devices. Each device has two addresses: Long address (64-bit globally unique ID) and short address 16-bit PAN specific address. On the other hand, IPv6 is widely deployed and desirable for the communication among small devices. It has a large addressing space that allows small devices to route IPv6 addresses. It requires 1280 bytes of MTU while 802.15.4 has an MTU of 127 bytes. It takes care of the fragmentation of the packets below the Network Layer and compresses the headers. 6LoWPAN header includes a Mesh address which enables routing in the network leaving the details of routing to the link layer (Ott 2012).

3.5.3 Zig Bee Stack

It is a logical stack at the network layer. It consists of three different characteristics: coordinator, router and end device. It is possible to have one coordinator in a Zig Bee network. The routers and the end devices are more dependent on the requirements of the applications and conditions. One important thing to mention is that within a network that supports sleeping end devices, the coordinator or a router should be assigned as a primary discovery cache device. This cache uploads and stores the discovery information and responds to discovery requests on behalf of the sleeping end devices. (Dynamic C 2008: 11-17.)

3.5.4 Transmission Control Protocol (TCP)/ User Datagram Protocol (UDP)

TCP/ UDP are located between the application and network layer. The basic task of TCP/UDP is to provide reliable, point to point transportation. TCP is responsible for addressing the data packet, the connection establishment and release, flow control, error handling etc. **Figure 15** shows the address header field.

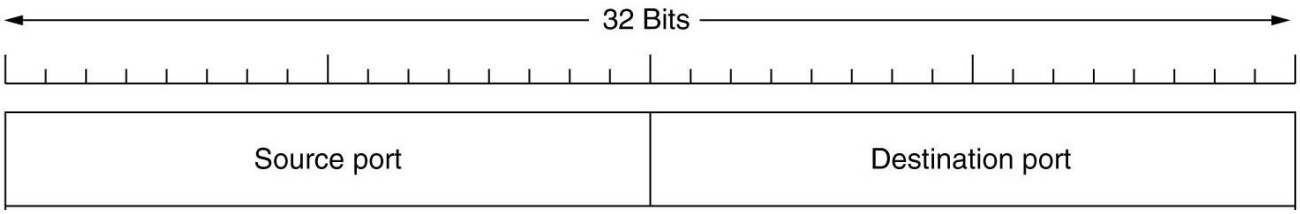

Figure 15. TCP Address Field (Fu 2008).

UDP destination port identifies the destination process, while UDP source port identifies the source process. It provides unreliable datagram services which includes the delivery of packets. The packets may be out of order or completely lost. There is no buffer register to store the information at the sender or receiver side because UDP takes care of the lost information. It is unreliable but a fast method. The total message length is in bytes includes the header and the data. Checksum is optional, 16 bit over header and data. (Fu 2008). UDP address header field is shown in **Figure 16**.

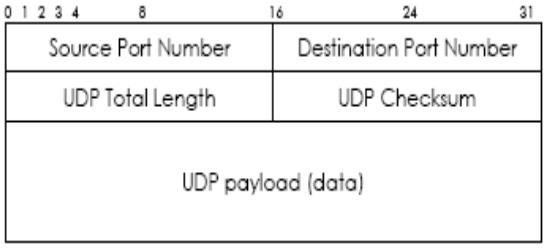

Figure 16. UDP Address Field (Fu 2008).

3.5.5 ICMP (Internet Control Message Protocol)

ICMP provides the facility for error reporting and IP support. Messages of ICMP are encapsulated as IP datagram. Requests can be sent to a host or a router which replies back to the querying host.

ICMP error messages report error conditions, usually it is sent when a datagram is discarded. Error messages are often passed from ICMP to application program. **Figure 17** shows the address header field of ICMP.

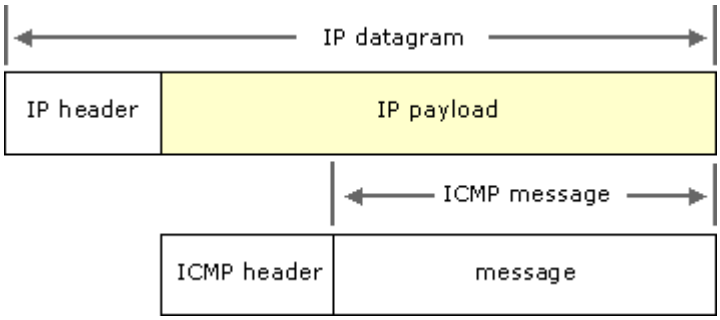

Figure 17. ICMP Address Field (Dynamic C 2008).

3.6 Security Issues in WSN

Security is one of the important issues in sensor networks. It can happen that an attacker tries to get illegal access to the unauthorized information, confidential data and system (Modares, Salleh & Moravejosharieh 2011). An attacker is a threat to the confidentiality of the data and can do the following:

- Alteration of the confidential data and ruin the data integrity
- Stealing of the material by being unauthorized intruder
- Disturbs and damages the characteristics of sensor nodes

WSN should be completely secure to be prevented from attacks and to provide the consistency and integrity of the network. Security requirements include the following:

- Authentication: The data can only be accessed by an authorized sensor node from a secure source
- Confidentiality: The mode of communication should be maintained secure when accessing the data contents.
- Integrity: It should be certain that the information has not been altered by an attacker. The received information should be the same as sent.

3.6.1 Encryption

Data in WSN must be secure. Encryption is one possible way to prevent the data to be cracked by intruders and other security threats. To achieve this task, information is transmitted over the medium by a cipher key which can only be known by the authorized person. It has a significant importance when it comes to the communication for military purpose or disastrous situations. This approach will minimize the attacks in the WSN and provide the security from any kind of intrusion.

4. SOFTWARE IMPLEMENTATION AND SERVER CONFIGURATION

4.1 Contiki Operating System

Contiki is an open source operating system often use for wireless sensor network. It provides all the necessary development tools such as Cooja simulator to run the WSN. Contiki supports dynamic loading and the replacement of programs and services. Contiki is built on an event-driven Kernel which can be used for multi-threading in multiple processes and individual processes. (Dunkels 2008.)

Contiki OS is developed for lightweight mechanisms and abstractions and capable to operate the system in constrained environments. Although, the OS is Kernel event-driven, it still supports multi-threading. Multi-threading is simply a library that links with those programs which requires multi-threading (Dunkels, Gronvall and Voigt 2004). Event-driven programming is often used in systems which include a number of memory constrains that fits in a general purpose operating system. An event-driven model does not support a blocking wait abstraction. So, in such case the programmer needs to implement the control flow, for a higher level logic by using a state machine. (Dunkels, Schmidt, Voigt & Ali 2006.)

4.2 Contiki System Architecture and Overview

Contiki has been ported to many microcontrollers such as Texas Instruments MSP430 and Atmel AVR. The MSP430 controller has 2 kilobytes of RAM and 60 kilobytes of ROM with 1 MHz processing power. These microcontrollers have a programmable flash memory on-chip. Contiki OS consists of a Kernel, libraries, loader, and set of instructions to perform processing (application

program or a service). A service represents the functionality of having more than one application process. Processes are dynamically replaceable during runtime. All communication between processes goes through the Kernel. The Kernel does not provide any hardware abstraction layer but it enables the device drivers to communicate with the hardware. (Dunkels, Gronvall and Voigt 2004.)

A process is defined by the functions of an event handler and the poll handler. A process remains in the private memory while Kernel points to the state of process. All processes share the same address space and cannot run in separated domains. Inter process communication can be done by posting an event. Contiki OS is classified into two parts: the core and the loaded program. A core consists of a program loader, the language run-time and support libraries and the communication stack with the device driver for hardware communication. A core is compiled as the binary image which is stored in the device before the deployment. The core cannot be modified after deployment. Programs are loaded in the program loader, it obtains the binaries of the program either using a communication stack or directly attached storage such as EEPROM.

4.3 Features of Contiki OS

Contiki system has some prominent features which are as follow:

- Loadable modules
- Threads: Preemptive threads and Proto-threads
- IP to Wireless Sensor Network
- 6LoWPAN: IP over 802.15.4
- Energy estimation and consumption through software

- Power consumption measurements at the network scale
- Communication in network shell
- Rime stack

4.4 Virtual Machine and Contiki Virtual Machine

The approach towards Virtual Machine significantly reduces the transmission of program code to a physical model. It also reduces the human efforts to deploy the actual machine because it provides a sophisticated environment in which the program code can run in a compact size. This approach is often used before the deployment of the WSN. There are a number of virtual machines available such as Java virtual machine which is well enough to perform such tasks. Most of the virtual machines are needed to be configured in order to run the desired application. A part of the program code is implemented on a virtual machine and the other part on the native code. (Dunkels et al. 2006.)

4.4.1 Contiki Virtual Machine

The design of the Contiki Virtual Machine (CVM) has been compromised for generic as well as for an application specific virtual machine. CVM can be configured, so that applications which are running on a machine can be implemented on the virtual machine and even without a virtual machine. The design of the CVM is the same as the other virtual machine such as Java. CVM is a stack based machine including separated codes and data areas. The instruction set includes integer arithmetic, unconditional & conditional branches and method invocations. Method invocations can be implemented in two ways: by invocation of CVM byte code functions and by invocation of functions implemented in native code. This instruction takes one parameter which identifies the

function to be called. The native function identifiers are defined at the time of the program compilation by the user. The user also compiles the list of native functions that can be called by the CVM program. By including the native function interface, the CVM program is able to call any native function including the services provided by the loadable program. In the CVM, the native functions are invoked just like any other function. The compiler uses the list of native functions to translate them into special instructions for calling native codes. Parameters are passed to the native functions through the CVM stack. (Dunkels et al. 2006.)

4.5 Architecture of the Kernel

The Contiki Kernel consists of a lightweight event scheduler which dispatches the events to run and periodically call the processes and the polling handler. The Kernel is not able to preempt an event scheduler once it has been scheduled, so the event handler must run till completion. There are two kinds of events in the Kernel: Asynchronous and synchronous event. Asynchronous events are procedures that call and queue an event to the target process. Synchronous events are the same as Asynchronous but they immediately cause the target process to be scheduled. Control can only be returned to the posting session when the event has been completed. Polling mechanisms are scheduled to a high priority synchronous event. A single stack is used by the Kernel to handle all the processes. The use of asynchronous events reduces the space of the stack as it occurs between each invocation of the event handler. (Dunkels et al. 2004.)

4.5.1 Loadable Programs

Loadable programs are implemented by using run time relocation function and binary format that contains the relocation information. When a program is loaded into the system, the loader first tries to allocate the sufficient memory based on the information provided by the binary format. Once the program is loaded into the memory, the loader calls the initialization function of the program. Initialization function may start/replace one or more processes.

4.5.2 Power save Mode

The Contiki Kernel does not provide any power saving abstractions, but it allows the applications to implement such mechanisms to operate specific parts of the system. Power saving is done by the help of the event scheduler which exposes the size of the event queue. This information is used to power down the processor. The processor can wake up by an external interrupt response. Poll handlers are used to handle these external events.

4.6 Conitki Libraries

The Contiki Kernel only provides the multiplexing and event handling. The rest of the part is optionally attached with the program and implemented as system libraries. Libraries are linked with the programs in three ways: Statically linked library program that is attached as part of the core, statically linked library program that is attached as part of the loadable program and program that can call the implementation of specific library service. These services can be dynamically replaced at run time. (Dunkels et al. 2004.)

Consider an example program which includes the following:

- memcpy() and atoi() functions are used to copy memory and to convert a string into integer
- memcpy() function frequently uses C library function
- atoi() function is less frequent and used seldom
- memcpy() function, in this case included in the system core
- atoi() function will not be included in the core system
- As the linked program starts to produce the binary format, memcpy() function will be linked against its static address in the core
- The object code as the part of C library which implements atoi(), must be included in the program binary format

4.7 Communication Support

Contiki allows the implementation of communication as a service to enable the run time replacement. This allows multiple communication stacks to be loaded at the same time. Moreover, the communication stack can be broken into different services like routing protocols and device driver which enables the run time replacement of individual parts of the communication stack (Dunkels et al. 2004).

Service mechanisms are used by the communication services to call each other and synchronous events, that enable the communication with the application programs. A device driver reads an incoming packet into the communication buffer and then calls an upper layer communication service by using the service mechanism. Contiki has two communication stacks shown in **Figure 18**: uIP and Rime. uIP usually called TCP/IP while Rime provides low overheads. An application is

able to use either one or both of them (Dunkels 2008).

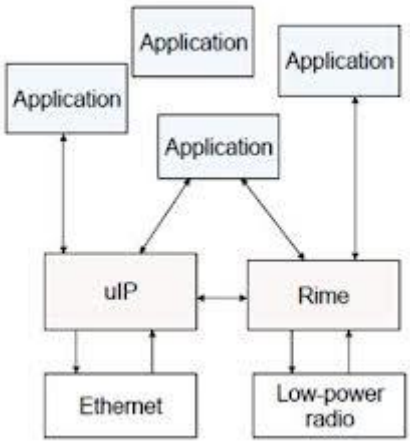

Figure 18. Communication stack (Dunkels 2008).

Characteristics of uIP:

- Declaration of processes, open TCP or UDP connections (tcp_connect(), tcp_listen(), udp_new())
- Establishing of new connections, post an event (tcpip_event) and connection is closed after the arrival of new data
- Process return, sending a reply packet
- Polling is done periodically at TCP connection
- uip_udp_packet_send() is the UDP packet send function. (Dunkels 2008).

There are two APIs used for uIP:

1. The “raw” uIP event-driven API. It used for small programs.
2. “Protosockets” (based on protothreads). It works better for large programs (Dunkels 2008).

Rime

- It is a lightweight primitive for a set of communication
- It composes both of the simple abstractions and more complex ones
- Rime is connected with a chameleon module, which adapts MAC layers, Data link layers and protocols (802.15.4 , 6LoWPAN, IPv6)
- Chameleon modules is responsible for header construction separated from communication stack
- Rime is responsible for packet attributes and deals with the communication stack including data collection protocols separated from the packet header (Dunkels 2008).

4.7.1 Communication Abstraction (Rime)

According to Dunkels (2008) there is a set of communication abstraction such as “Single hop broadcast (broadcast), Single hop unicast (unicast), Reliable single hop unicast (runicast), Best effort multi hop unicast (multihop), Hop by hop reliable multi hop unicast (rmh), Best effort multi hop flooding (netflood), Reliable multi hop flooding (trickle), Hop by hop reliable data collection tree routing (collect), Hop by hop reliable mesh routing (mesh), Best effort route discovery (route-discovery), Single hop reliable bulk transfer (rudolph0), Multi hop reliable bulk transfer (rudolph1).” **Figure 19** presents the block diagram of the Rime communication stack in which each module reduces complexity by compiled code between 114 and 598 bytes. The complexity is handled through different layers.

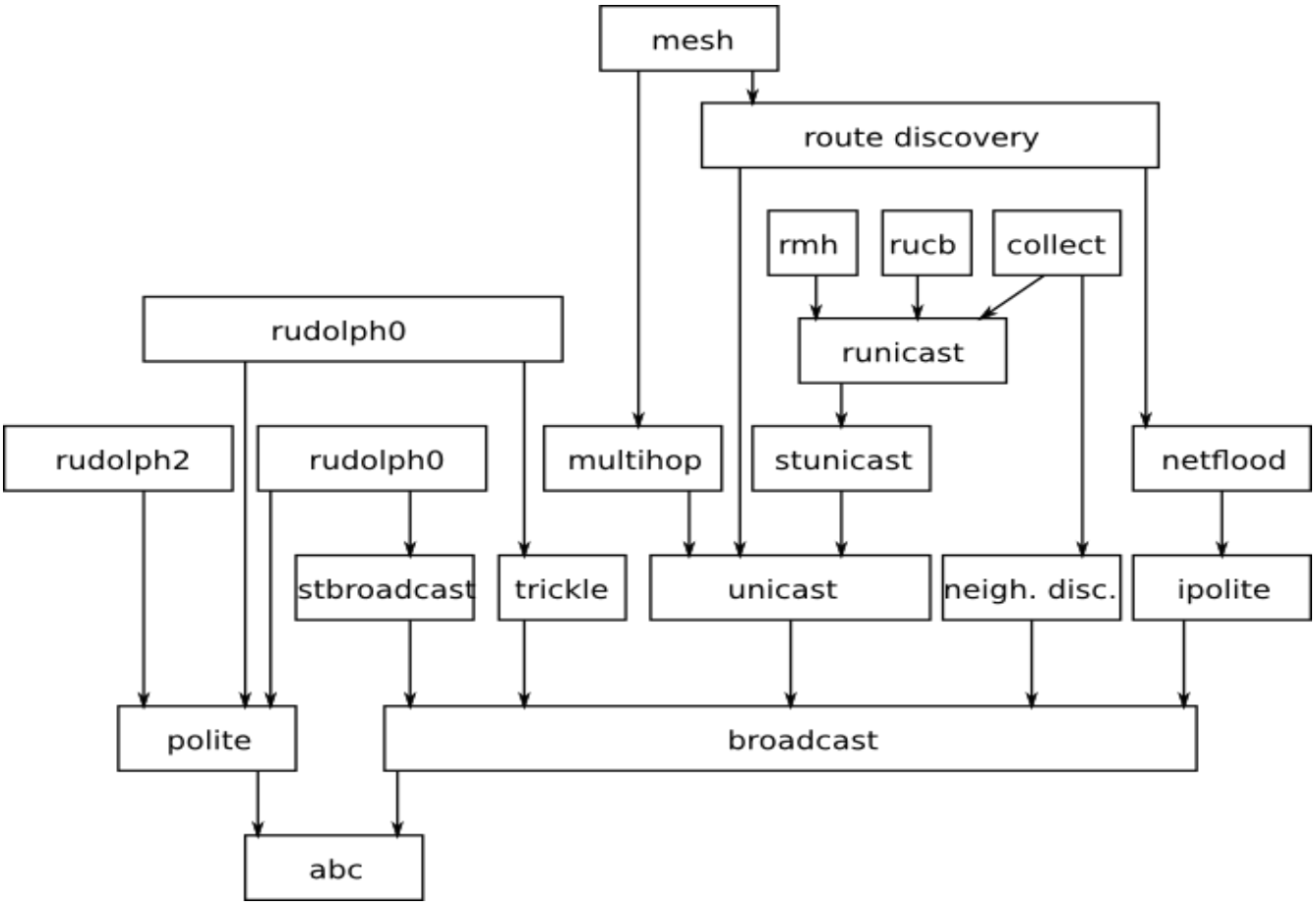

Figure 19. Rime Communication stack (Dunkels 2008).

Rime communication is identified by 16-bits channel. The nodes are responsible to choose which module to use on a particular channel. For example: unicast on channel 155 and net flood on channel 130. Channel numbers smaller than 128 are reserved by the system, the shell or some other applications. When a packet arrives, a time out or an error condition occurred before the Callback is initialized. The connection must be opened and the arguments should include: (module structure, channel, callbacks).

Let's take a look at an example how the callback works when a message (data) is received. (Dunkels 2008):

```
void recv (struct broadcast_conn *c)
{
printf(" received message ");
}
```

The above method called, when data is received. To view the message, callback must be initialized:

```
struct broadcast_callbacks cb = { recv};
struct broadcast_conn c;
```

After this, we will open the connection by setting up the arguments as discussed above:

```
broadcast_open (&c, 128, &cb);
```

- &c is pointing to the connection.
- 128 channel setup.
- &cb is callback structure.

We can also send this message by initializing the method:

```
void send_message (char *msg, int len)
{
rimebuf_copyfrom (msg, len);
broadcast_send (&c);
}
```

So far, the discussion provides the details of the Contiki operating system, the architecture of Kernel, Processes, concepts of Contiki Virtual Machine and the communication stack. It can be seen that Contiki is useful for the deployment of small wireless sensor networks and also used for multiplexing the hardware of a sensor network with multiple applications. Next we will see the application development with Java using JDBC, JNI and GUI.

4.8 Application Development with Java

4.8.1 Java Programming Language

Java is a class based, object oriented, general purpose programming language which has few implementation design dependencies. Java is an independent platform application which can be written on one platform and run on many platforms. Java codes compile on virtual machine which makes it platform independent. Java is the first language which is capable of creating network applications and dynamic web pages (applets). Java offers abstraction, exception handling, garbage collection and libraries loaded with networking protocols such as (TCP/IP and FTP). Java also provides Graphical User Interface (GUI) which can run on different OS without changing interface. Some of the contents of Java that we are going to use in the thesis are:

1. JDBC (Java Data Base Connectivity).
2. JNI (Java Native Interface).
3. Java GUI (Graphical User Interface).

Next we will discuss one by one the characteristics and functions which are used.

4.8.2 JDBC

An important area of the Java development is to build software which is capable of accessing the database. JDBC provides an interface through the Java software which is interactive, user-friendly and capable for the database connection. It allows SQL access and retrieval to display a database on Java application or over the Internet. Java is one of the leading development environments to create database applications by using the functionalities of the library as application programming interface (API) and to be specific, Java Database Connectivity (JDBC). (Guan, Horace &Zhang 1998.)

Java Database Connectivity is an API specially used to execute SQL instructions. It contains a set of classes and interfaces which are written in Java programming language. JDBC allows writing database applications through Java software. It capable to access the data stored in tables in a relational database system. JDBC uses a relational view of data and SQL to access them. It is actually made on the top of the Open Database Connectivity (ODBC) standard which is a commercially available database (Raimund 2000). Java application connects to the database through JDBC API.

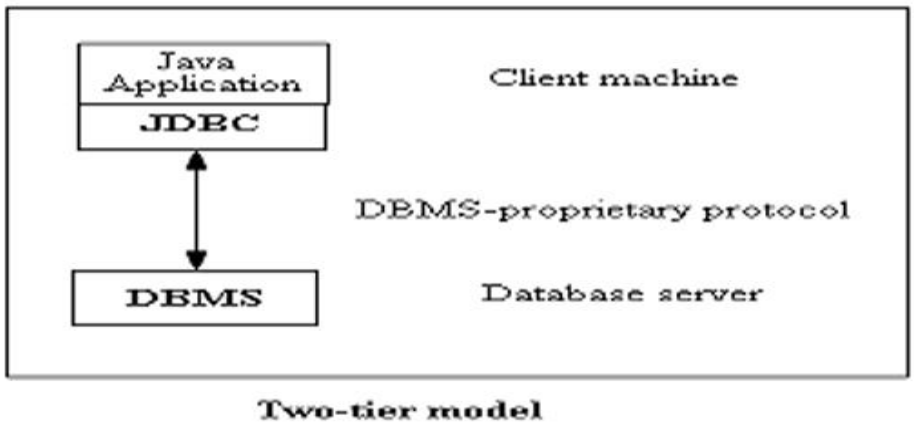

Figure 20. Two tier model for JDBC connection with database (Raj 2012).

The Structured Query Language (SQL) is the most commercial common language used by Database Management System (RDBMS) to provide complete mechanisms of issuing the SQL instructions. **Figure 20** shows the interface of the JDBC driver. In order to access the contents of the database, JDBC has to execute SQL query, which will access the database and retrieve the desired contents to appear on the Java application.

Figure 21 elaborates the interface and connectivity from the Java application through the Database Management System. The three tier models are also used for the connectivity but the concept will remain the same except for the JDBC-ODBC bridge. JDBC-ODBC is a bridge which provides a gateway interface from JDBC to ODBC drivers. Some of the database packages come with ODBC drivers such as MS Access.

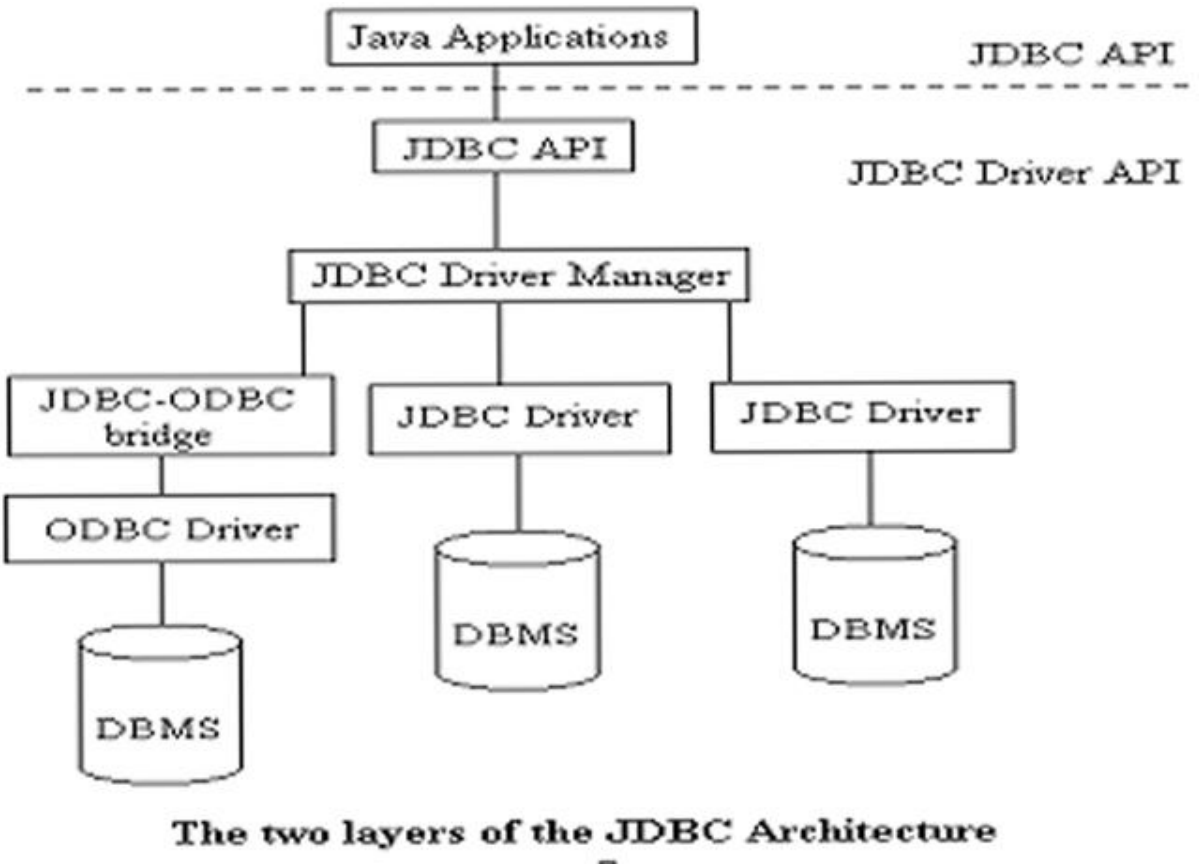

Figure 21. The actual connectivity of JDBC (Raj 2012).

In the following pages, the flow charts for establishing and closing connection, as well as creating a table and inserting values with JDBC are presented:

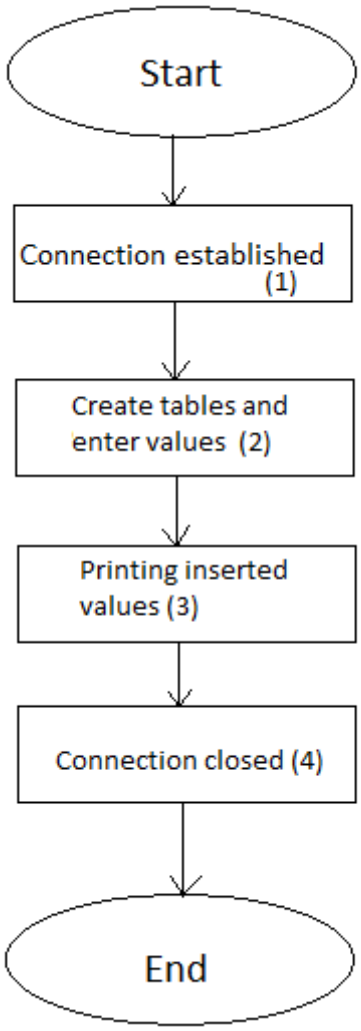

Figure 22. Flow chart of the database connectivity (general overview).

Figure 22 explains in detail of how the connection established, conditions of connection establishment and conditions for inserting a table values. Each block is explained in the flow chart in **Figure 23**.

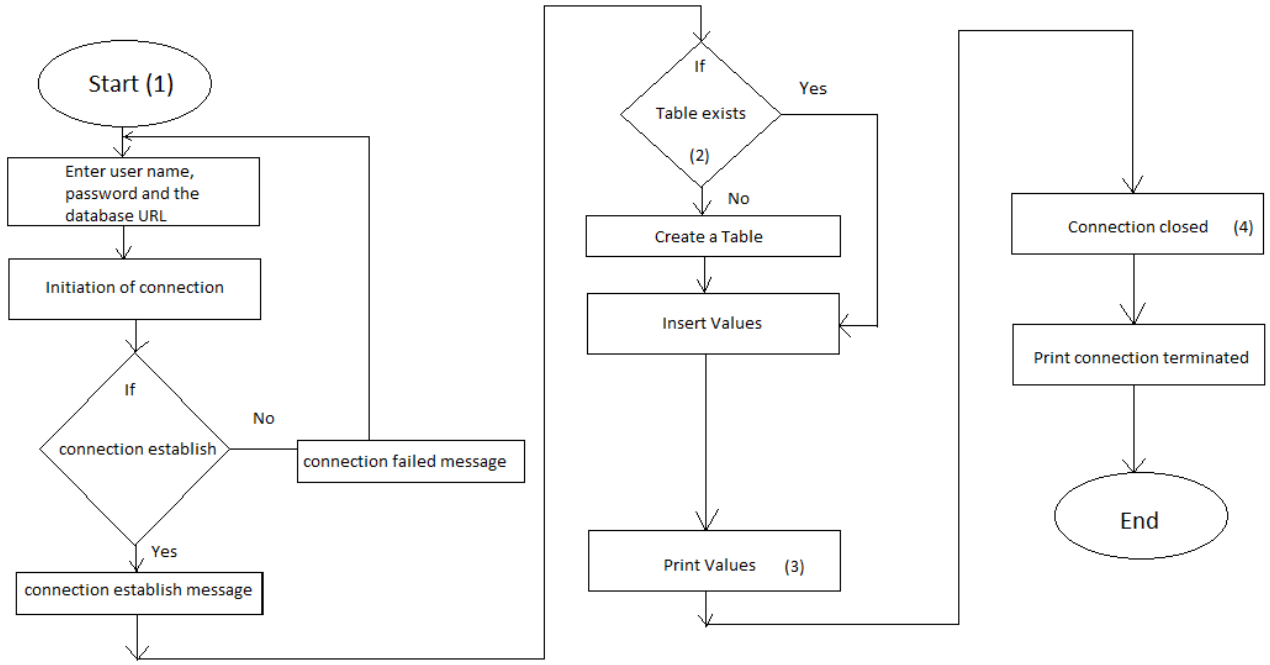

Figure 23. Flow chart of the Database Connectivity.

Codes for connection establishment are as follows:

String userName = "pma"; //saving user name in a variable

String password = ""; //saving password in a variable

String url = "jdbc:mysql://localhost/test"; //jdbc:mysql is syntax to access URL

Class.forName ("com.mysql.jdbc.Driver").newInstance (); //initiating connection

conn = DriverManager.getConnection (url, userName, password); //initiating connection

System.out.println ("Database connection established"); //prints when connection established

The following program part shows how the values of the temperature, accelerometer, battery, time and date are entered to the database.

/*Inserting table**********/

```
Statement s = conn.createStatement ();
int count;
s.executeUpdate ("DROP TABLE IF EXISTS data");
s.executeUpdate (
        "CREATE TABLE data ("
        + "id INT UNSIGNED NOT NULL AUTO_INCREMENT,"
        + "PRIMARY KEY (id),"
        + "Temperature CHAR(40), Accelerometer CHAR(40),"
        + "Battery CHAR(40), Time CHAR(40), Date CHAR(40))");
  System.out.println (count + " rows were inserted");
```

The above codes provide the SQL statements to access the database. At the same time, it also inserts a table into the database and confirms once the rows have been inserted. This data can be retrieved in a systematic way can be printed to the screen with the Java application.

4.8.3 Java Graphical User Interface (GUI)

Java GUI is an application program which interacts with the user. It also provides the capability of entering data which contains certain functionality and returns the results. A Java GUI includes different package libraries. The ones that have been used are: java.awt and java.swing. These libraries are responsible for making Buttons, Labels, Text Fields, Text Areas, Check Boxes, Radio Buttons, and Drop down List, Panels and Layout Manager. There are also some events involved to perform some action on the GUI.

4.8.4 Java Native Interface (JNI)

When it's necessary to overcome the memory management and performance constraints in Java, native codes C/C++ should be used. Java also supports native codes by the Java Native Interface (JNI). JNI can be a little bit complicated because of using two different languages but very useful applications can be made by using JNI. Code for the Java class that uses JNI:

```
public class JavaJNI {                          //main class
   static {
   System.loadLibrary("NativeLib");             //Loading library
   }
   public static native int open(String devName);   //Constructors
   public static native void conf(final int fileDescr);
   public static native String read(final int fileDescr);
   public static native void close(final int fileDescr);

   public static void main(String[] argv) {     //main function
     int fd = open("/dev/ttyUSB0");             //opening USB port
   JavaJNI.conf(fd);                            //setting up configuration
   System.out.println(JavaJNI.read(fd));        //printing the data read
   JavaJNI.close(fd);                           //closing the port

                  }
   }
```

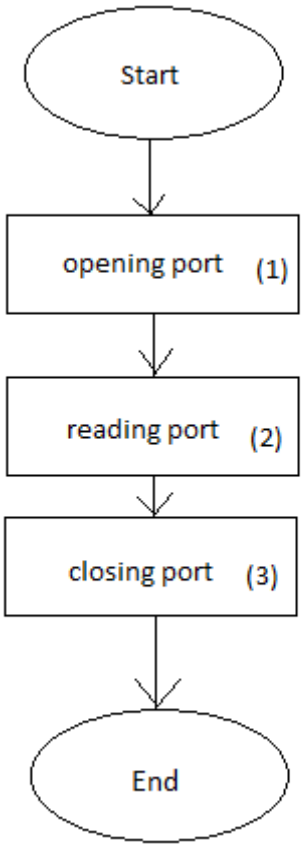

Figure 24. Flow chart of the Java Native Interface (general overview).

In **Figure 24**, the configuration of the serial port is handled in the opening part. The following flow chart will explain the above process in detail. Each individual blocks are explained in **Figure 25**.

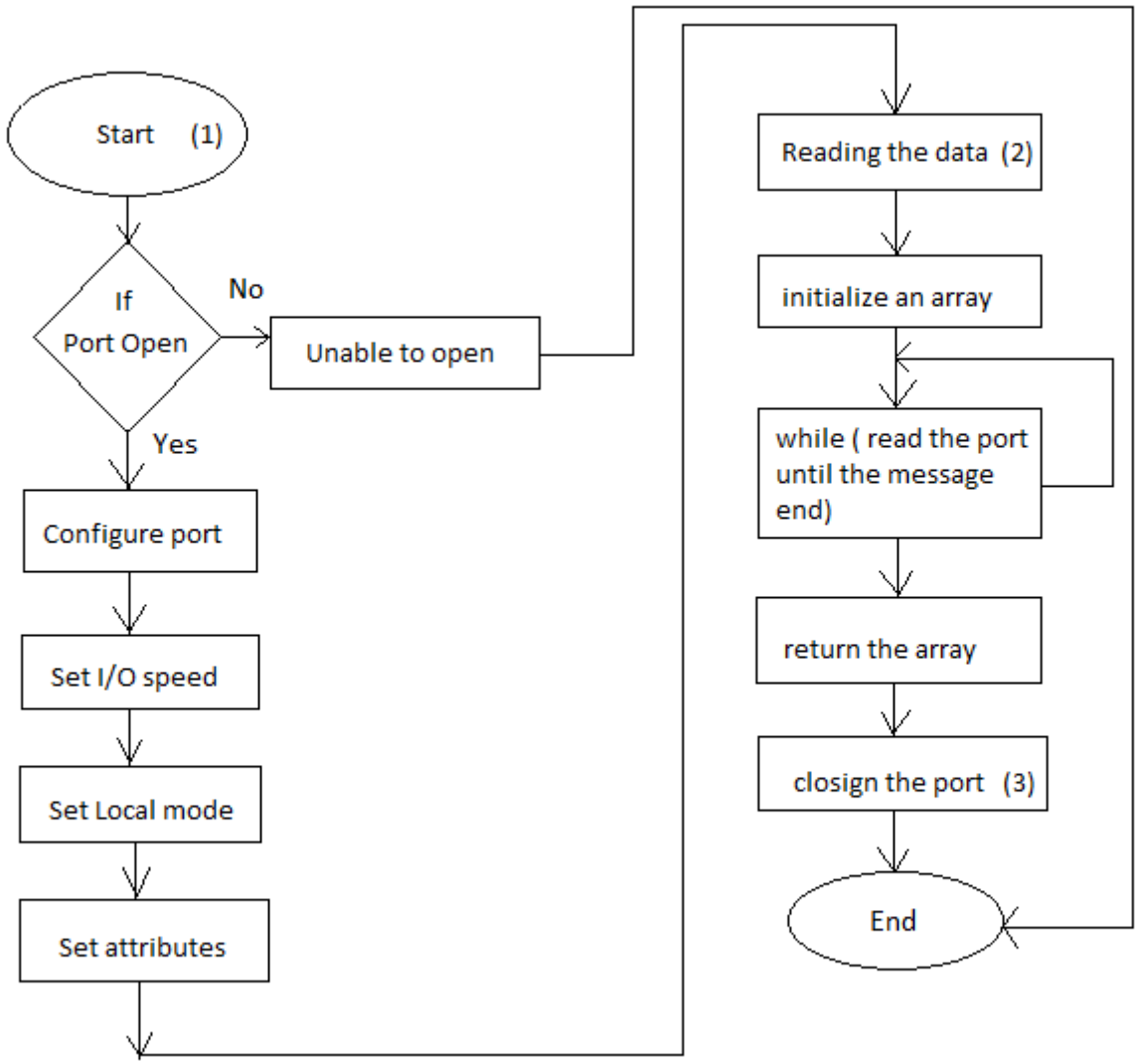

Figure 25. Java Native Interface.

4.9 Website Development

The Website Development is an essential requirement in the thesis to provide the information (data from wireless sensor network) on the Internet. To achieve this, we are using LAMP stack which is free open source software. LAMP provides a combination of Apache HTTP server, MySQL and PHP for a Linux environment. The software combination has become very popular because it is free of cost, open source and because of its components which are compatible with most current Linux distributions.

To perform the prototype experiment, a local host web server has been installed. In the following, each software which took part in the website development will be discussed along with the codes.

4.9.1 PHP

PHP (Hypertext Preprocessor) is a server side scripting language which offers a general purpose programming mode as well as web development design. PHP codes are interpreted by a web server with a PHP processor module which generates the web pages. PHP can easily be integrated in the HTML scripting source without needing any external file to process.

The purpose of using PHP in this thesis is particularly related to website development which is secure and which provides compatibility. PHP provides the means to fetch the data from the database easily and makes it appear on the website. It also provides the means of using logics to make a good result. To make the website more interactive and colorful, CSS can be used. Cascaded Style Sheet (CSS) is used to make an appealing website (Wikipedia 2013 c).

In the next step an HTML code is introduced.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3c.org/1999/xhtml" xml:lang="en" lang="en">
<head>
    <link href="css/style.css" rel="stylesheet" type="text/css" />      //Note that CSS is used
    <meta http-equiv="Content-Type"content="text/html; charset=utf-8"  />
    <title>Data Clollecting Center </title>
</head>
<body>
        <div id ="header">
                <h1>Cloud Platform</h1>
        </div>
<div id="container">
        <p class ="title">Welcome to the Data Center</p>
                <div id="navigation">
                        <br/><br/><br/><br/>
                        <a href="main.html">Home Page</a>
                        <a href="">Next Page</a>
                        <a href="">Set Values</a>
                        <a href="">About</a>
                </div>
                <?php
                        //Here we implement our PHP logic to get the data
```

```
                    ?>
                    <div id="content"><!--Data and Graph.-->
                            <h1 class="title">Temperature Graph</h1>
                            <p>This data is taken from the WSN.</p>
          </div>        <!-- End of the page-->
        <div id="footer">
                    <p>Copyrights &copy; <a href="#">Plain and simple</a>Designed by
<a href="Kamran">Kamran</a> </p>
        </div>
</div></body></html>
```

The above HTML code is self-explanatory, carrying a very simple structure to build a website. PHP is an Internet aware system with modules built in to access FTP servers and database servers. PHP also allows designers to write the code in C language and add its functionality to the PHP language. PHP is especially used at the server side web development which runs on a web server. Any PHP code can be executed by PHP runtime which usually creates a dynamic web page content like images on website. (Wikipedia 2013 c.)

Security in PHP applications is necessary to prevent the website from intrusions. For this purpose, PHPIDS provides security features to the PHP application such as detection of the attacks based on cross site scripting, SQL injection, header injection, directory transversal, remote file execution and inclusion, and denial of service (DoS).

4.9.2 MySQL

MySQL is world's most used open source Relational Database Management System (RDBMS) that provides an interface that accesses a number of databases. When it comes to web application development, MySQL is the best choice to make query transactions to the database. It is the central component of LAMP which is necessary to store the data in the database and to retrieve the data from the database. MySQL does not offer administration tools to manage the data within MySQL database. Users are able to use command line tools or desktop web applications which can create and manage MySQL database, build database structures and back up the data. (Wikipedia 2013 d).

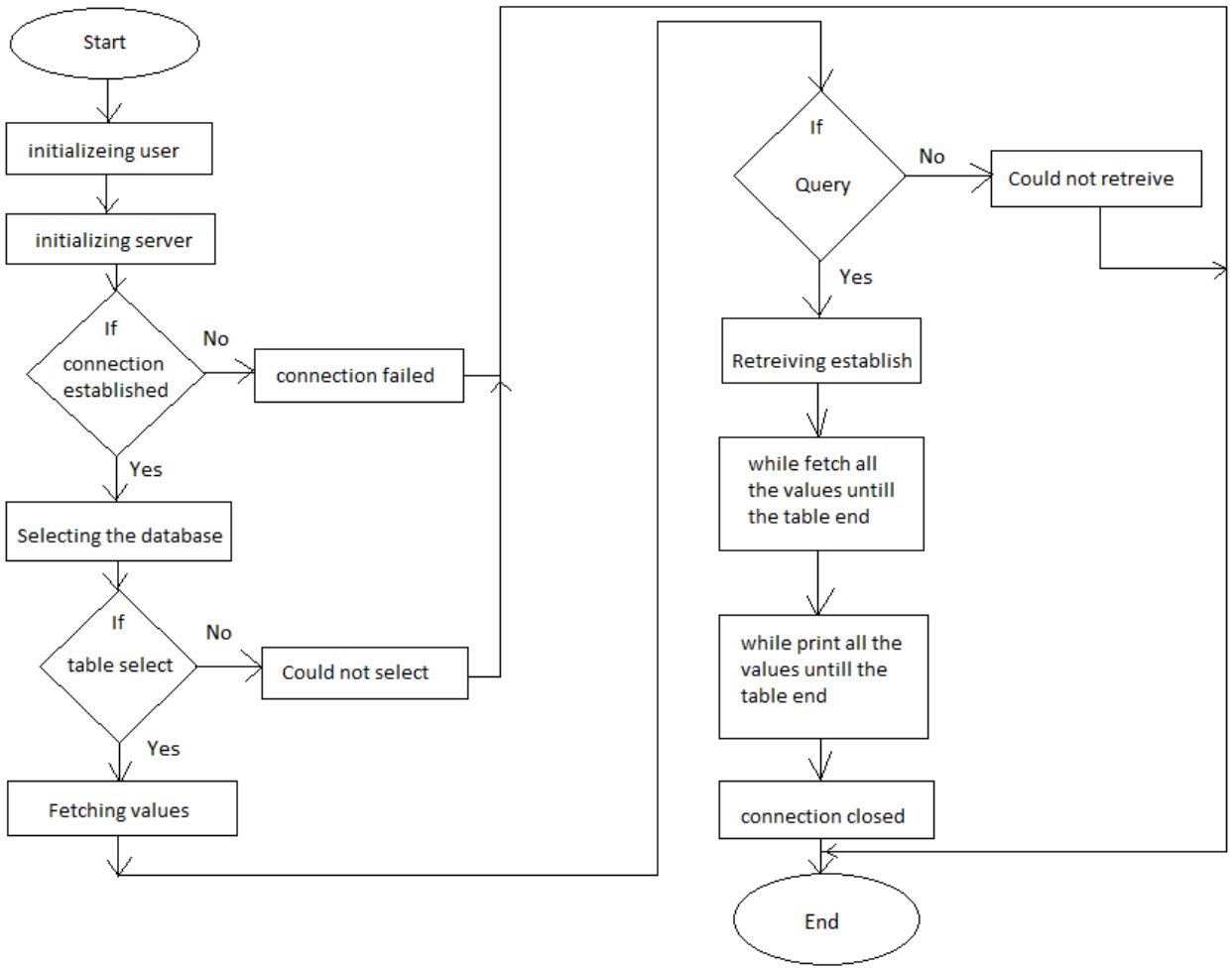

Figure 26. MySQL.

Figure 26 explains the flow of MySQL connection. These codes show how to fetch the data in PHP using MySQL statements.

```
$user = 'pma';                  // Here we define connection values
$pswd = '';
$dbName = 'test';
$server = 'localhost';          //Here we try to establish the connection
                                //We also use error control operator @ to suppress error and warning
                                //messages
 $dbConn = @mysql_connect($server,$user, $pswd);
if (!$dbConn)
        {
    echo "Connection failed: " . mysql_error() . "<br/>";
        }
        else
        {
                if(!mysql_select_db($dbName))           //Here we select a database.
                {
                        echo "Could not select $dbName database: " .
                        mysql_error($dbConn);
                }
                else
                {
                 // select the table
                 $table = 'data';                       }} // Table & Database selected
```

This part shows how to fetch the data from the database.

```
$queryString="select * from $table";
$queryResults=mysql_query($queryString, $dbConn);
if($queryResults == FALSE)
{
        echo "Couldn't retrieve data from $table<br/>";
}
else
{
        $result = mysql_query($queryString, $dbConn) or
        die(mysql_error());
                while ($i < mysql_num_fields($result))
                        {
                                $meta =
                        mysql_fetch_field($result, $i);
                                if($i>0)
                        $header[]=$meta->name;
                                if (!$meta)
                                {
                echo "No information available<br />\n";
                                }
                                $i++;
                        }
                echo"<br/><div><table><tr>";
```

```
echo"<td >$header[0]</td><td>$header[1]</td><td>$header[2]</td>
<td>$header[3]</td><td >$header[4]</td></tr>";
//Here we read data into an associative array and iterate automatically through all
//rows.
while(($row=mysql_fetch_assoc($queryResults)) !== FALSE)
                {
                        foreach($row as $columnName=>$value)
                        {
                                if($columnName != "id")

                                echo "<td>$value</td> ";
                        }
                        echo"</tr>";
                }       //end of while
                        //end table and div.
        echo "</table></div>";
        }       }
mysql_close($dbConn);           // In the end close the connection
```

The above code is used to fetch the data from the database. After that, all the data and values are displayed on the web page.

4.9.3 Apache HTTP Server

Apache HTTP is a web server software program which was developed to use on operating system. Apache server offers developing and hosting web sites with secure authentication schemes. Its authentication and security modules include: mod_access, mod_auth, mod_digest, mod_auth_digest and the successor to mod_digest. Secure Socket Layer and Transport Layer Security support (mod_ssl), a proxy module (mod_proxy), a URL re-writer (mod_rewriter), custom log files (mod_log_config) and filtering support (mod_include) and (mod_ext_filter). Wikipedia (2013 e)
Apache provides a compression method of external extension (mod_gzip), which is implemented to reduce the size of web pages over HTTP. Another open source intrusion detection and prevention for web applications is Mod Security. Apache serves many websites by bringing the features of configurable error messages, authentication of database and content negotiation supported as Graphical User Interface (GUI). (Wikipedia 2013 e.)

4.10 Server Configuration

Ubuntu server has the same archive as the standard Ubuntu operating system distribution. It also installs a set of distinctive default packages. These packages are very small because the installer will not install a graphical environment or programs like in the standard Ubuntu OS. All Ubuntu server packages are brought from the same archive of the official Ubuntu so that the user is able to install it later. The archive gives you the flexibility to transform from Kubuntu to a running Ubuntu server. The ground values have been established so that the Ubuntu server has different set of packages than the ordinary Ubuntu. (Hill, Helmke & Burger 2010: 141-178.)

4.10.1 Differences in Ubuntu Server

The most important difference is a custom server Kernel. The kernel employs the internal timer frequency to 100Hz rather than 250 Hz in ordinary Ubuntu. According to Hill, Helmke & Burger (2010) "[The] I/O scheduler is used instead desktop CFQ scheduler, a batch of other minor tweaks for virtualization, memory support and routing". The goal is to provide extra ordinary performance and throughput for server applications. Furthermore, it supports basic NUMA which is a memory design used in multiprocessor systems which drastically increases the multiprocessing speed.

4.10.2 Steps for installation and configuration

Step 1

First step is to install Ubuntu server. It is open source free software which can be obtained from the Internet. It can be installed by using a CD or memory stick.

Step 2

After the server installation, there are few steps that are recommended to follow:

Package Management

Ubuntu offers different package features like system management, upgrade and configuration. According to Hill, Helmke & Burger (2010) "[Ubuntu] archive has five repositories" which are the following:

1. Main: This repository installs packages by default which has official support

2. Restricted: It includes software with restricted copyrights mostly hardware drivers
3. Back ports: It offers newer version of packages provided by community
4. Universe: It contains packages maintain by Ubuntu community
5. Multiverse: It offers software with some price

APT-Get Repository

APT stands for advance package tool which is a powerful command line tool with functions such as installation of new software packages, upgrading packages and upgrading the entire Ubuntu system. This tool provides the ease to the user working over simple terminal connection (SSH) and system administration scripts. (Hill, Helmke & Burger 2010.)

Aptitude

Aptitude is best suited for non-graphical user interface environment which ensures proper functioning of command keys. It is the highest level of the package management stack a neat and colorful textual front end that can be interchangeable with apt-get.

Configuration

Configuration of the apt system repositories stored in the /etc/apt/source.list configuration file. The repositories can be entered or deleted by the users with admin rights.

Step 3

After Package management, there are also some other configurations which include Network Configuration (TCP/IP, DHCP), Domain Name Service (DNS) installation & configuration, Remote Administration (Open SSH, eBox), Network Authentication (Open LDAP Server) and Web Server configuration.

Security

One of the important tasks for a system Administrator is to deal with the server security. Server security becomes more important when it is connected with the Internet. Ubuntu server is itself a very secure platform. The team produces official security updates. Ubuntu has no open port policy, which means once you install Ubuntu server on your PC, it will not allow any software to get access from the Internet by default.

Account Administration

Ubuntu does not provide the root or administrator account by default which improves the security. The user added during installation process is by default placed into the admin group and may use sudo to perform administrator tasks. Sudo also permits a secondary user to execute some commands with super user privileges. It also allows the administrator to add & delete users and secures the user profiles and password policies.

Firewall

The Linux kernel includes Net filter subsystem. It is used to measure the data traffic in the network received or sent through the server. It also maintains an IP table, so that when a packet is received by the server, it will allow the Net filter to take decision whether to accept or reject it depending on the rules supplied by the user space via IP tables. This IP table is required to manage firewalls.

Certificates

The most common forms of cryptography is the public-key encryption. This phenomena work through a combination of public key and a private key. When a system encrypts information with the public key, it can only be decrypted using private key. The applications of encryption can be seen in Secure Socket Layer (SSL) and the Transport Layer Security (TLS) connection. A Certificate is a method used to distribute a public key and other information about a server and the organization that is responsible for it. The Certificate can be digitally signed by a Certificate Authority (CA) which is a third party assurance. It is responsible for the information inside a certificate.

To achieve total security, requires an incredible deep and inner working knowledge of computer systems. No system is completely secure. Securing the system does not mean just to prevent the system from all of the attacks but also to make it difficult for the attacker, so that it is not worth to break in the system.

5. EXPERIMENTS & GRAPHICAL USER INTERFACE (GUI)

In order to integrate the WSN to the website, a Graphical User Interface (GUI) application has been developed. This application allows the user to receive the sensor data and automatically upload this data to the database. The application is developed with Java. The Java Native Interface (JNI) has been used to get the data from the WSN. For the GUI the Java Swing library has been used. To communicate with the database the Java Database Connectivity (JDBC) API is used. All these attributes are combined in one application shown in **Figure 27**:

Figure 27. Graphical User Interface (GUI) Application.

5.1 RSSI Measurements

This experiment was done in the corridor of Technobothnia to measure the RSSI values between two wireless sensor nodes at multiple distances. The measurements were taken by exchanging a message from one sensor node to another and the resulting RSSI values were taken on the laptop which is connected to one of the sensor node.

First the RSSI values were measured on the ground between two sensor nodes. **Figure 28** elaborates the behavior of the RSSI values at different distances. 10 RSSI values were measured at each distance and the average RSSI value has been taken.

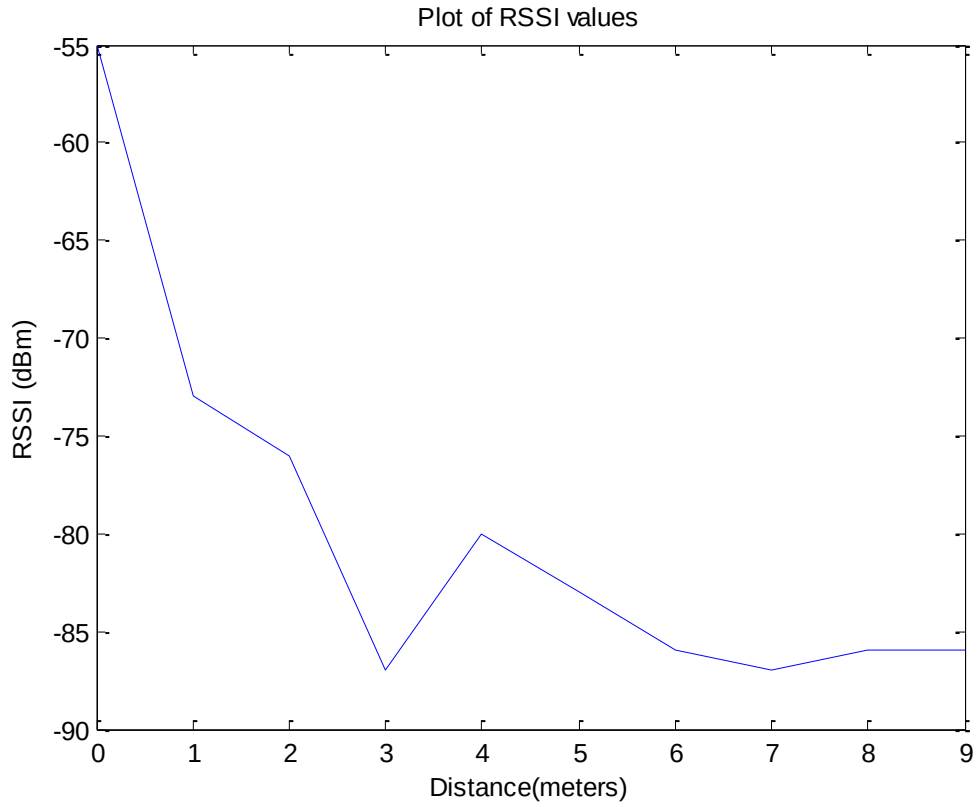

Figure 28. Measured RSSI on the ground between two sensor nodes.

As the distance increases, the RSSI value decreases. Furthermore, the RSSI value at the distance of three meters is minimum which depends on the distortion between the sensor nodes and the environment conditions. It can be seen from distance one to three meters, the plot drastically decreases which shows sensitivity of the signal strength of the sensor node. The estimated values are fairly good from four meters to nine meters which indicates that the relation between RSSI and the distance is linear.

Secondly the measured RSSI values were taken above the ground between two sensor nodes. **Figure 29** shows the behavior of RSSI measurements at different distances.

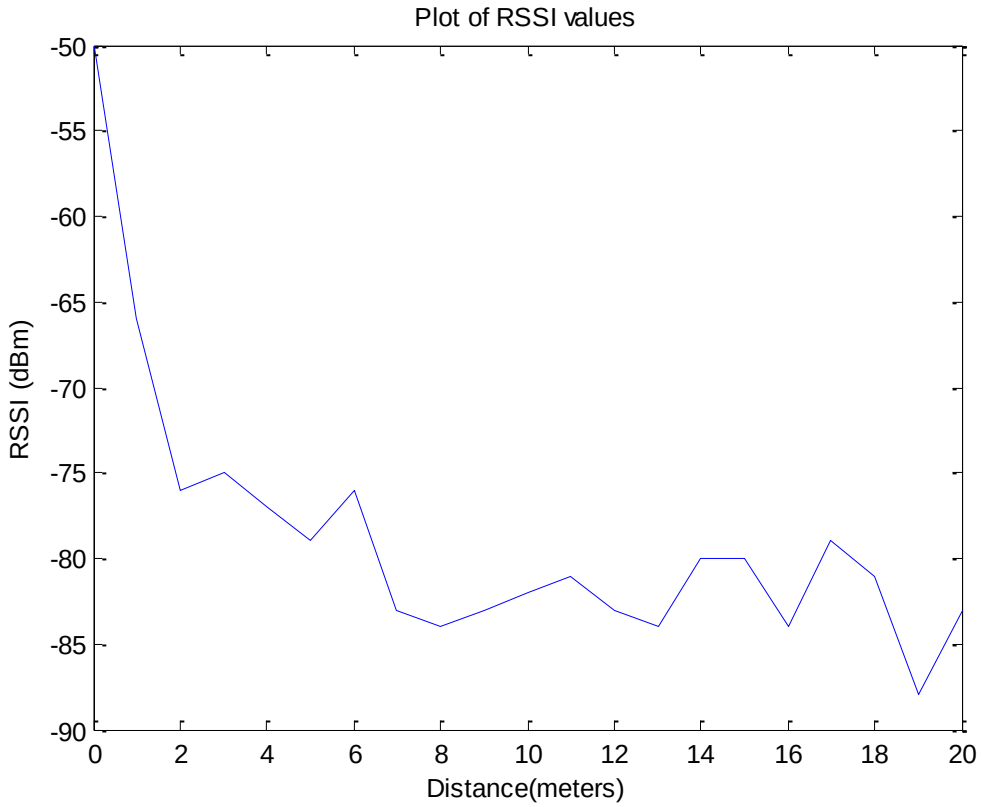

Figure 29. Measured RSSI between two sensor nodes above the ground.

The plot indicates the inversely proportional relation between the RSSI and the distance. It can be seen that the RSSI measurements are almost linear for the distances in the range between one and twenty meters. It is noticeable that the RSSI is high at six meters and is low at 16 and 19 meters which is dependent on the construction of the building and the environmental conditions. The plot guarantees 20 meters of communication between two sensor nodes inside a building.

Figure 30 compares the RSSI values measured on the ground and above the ground. The measurements show that the RSSI values above the ground is higher than the RSSI values on the ground. The reason is that ground absorbs the power transmitted from the sensor nodes.

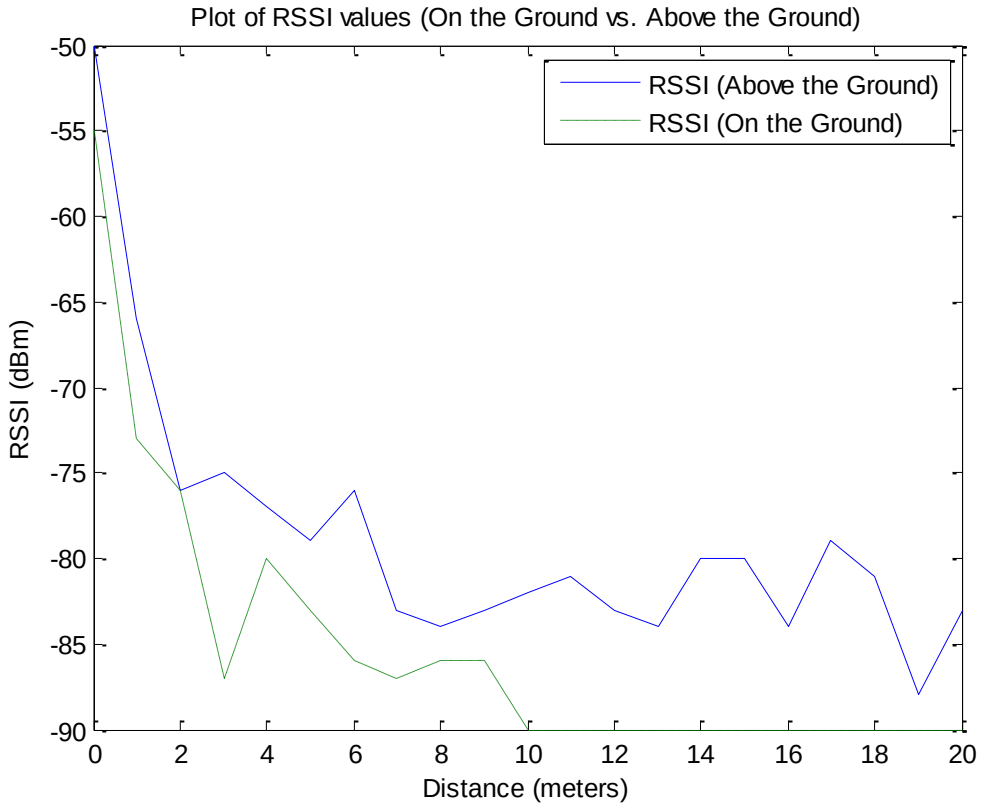

Figure 30. RSSI measurements on the ground vs. RSSI measurements above the ground.

5.2 SQL Query Execution Time Measurements

In this part of experiment, the query execution time is presented. The measurements have been taken using PHP in which the transfer time and the total time duration of executing one query in SQL is shown. **Figure 31** presents the time required to execute one query at different time intervals. The values have been taken using the local web server at different time intervals. The execution time varies in every time interval when executing a query. It is important to notice that the execution time depends on the load of the sensor.

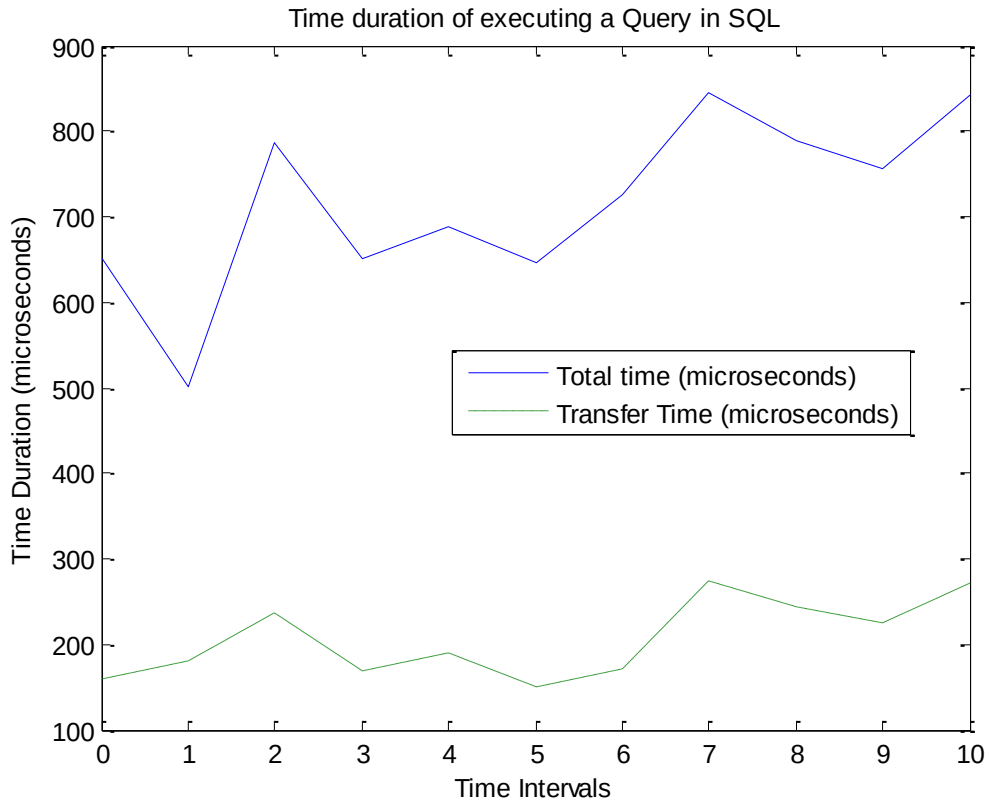

Figure 31. SQL Query execution time.

The total time duration is measured by the summation of different time values given in **Table 1**.

Table 1. Time duration measurements of different factors involved in the execution of a query.

Factor	**Time duration (microseconds)**	**Factor**	**Time duration (microseconds)**
Query starting	58	Checking permissions	16
Opening tables	51	System lock	21
Optimizing	12	Preparing	22
Executing	9	Sending data	166
End	16	Query end	15
Closing tables	16	Freeing items	32
Logging slow query	10	Cleaning up	10

The different factors involved in the execution of one query are mentioned in **Table 1**. The total time of execution can be calculated by simply adding all the above values plus the transfer time.

5.3 Security Measurements

Security is the most important part in web services. To evaluate the performance of the security, the average processing time for encryption has been measured. The processing time has been measured using the PHP script of two known algorithms: MD5 and sha512.

MD5 is a cryptographic hash function which process as the messages with different length into a 128-bit fixed length hash value. This hash value is then broken into 512-bit blocks of chunks, so that the length of the message is divisible by 512. Just like any other encryption technique, it also sends the message by encrypting it with public key and decrypts it with the private key.

The sha512 algorithm requires a number of operations in the mixing function and it works on 64-bit words. It requires only one block of operation to perform 80 iterations. Sha512 is faster in most of the practical message sizes and one of the strongest algorithms. It also uses a public key to encrypt and a private key to decrypt the message.

Figure 32 illustrates the behavior of the processing time. The plot indicates that the processing time is almost the same from 0 byte to 2 bytes of input data but it is drastically increases with an input of size bytes. The processing time fluctuates to the maximum time at 20 bytes of input. This result shows that the processing time is proportional to the number of bytes at the input side and it starts to increase as the number of bytes increase.

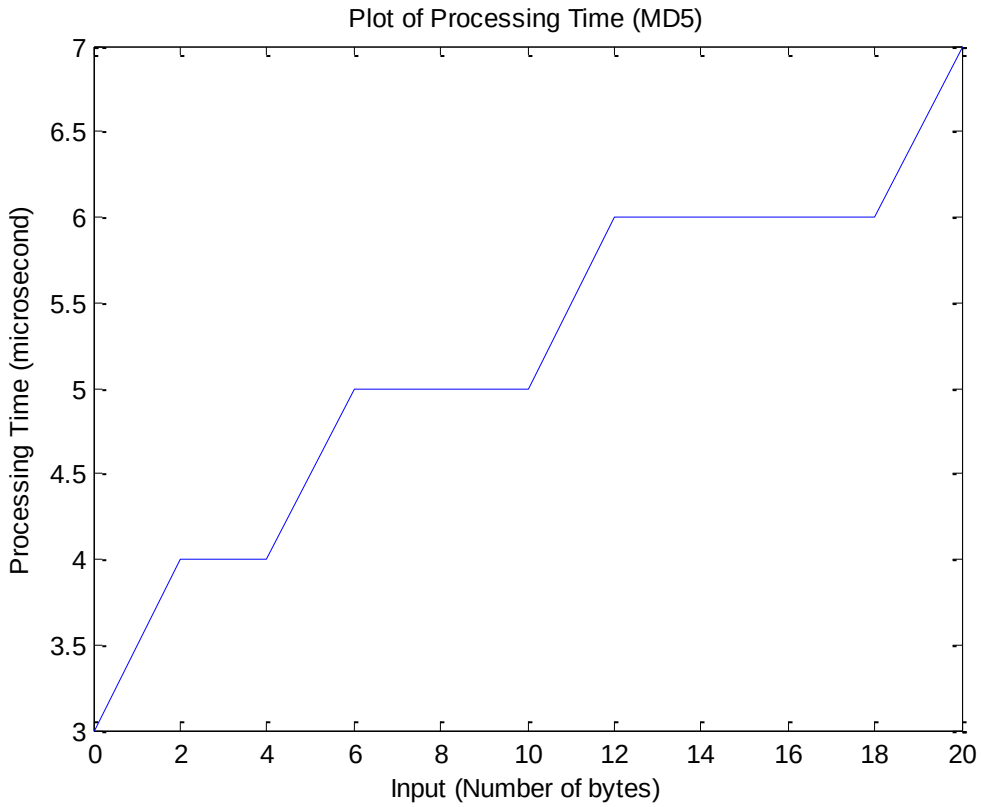

Figure 32. Plot of Processing Time (MD5).

Next the processing time of sha512 algorithm is shown in **Figure 33**. The plot indicates that the processing time in sha512 is relatively high. It has the minimum value of the processing time from 0 byte to 4 bytes. But it is increases to the minimum value as the number of bytes at the input side becomes 8 bytes. Processing time is almost the straight line from 14 bytes of input data to 20 bytes which indicates that sha512 algorithm has the high processing time as the number of bytes increase at the input side.

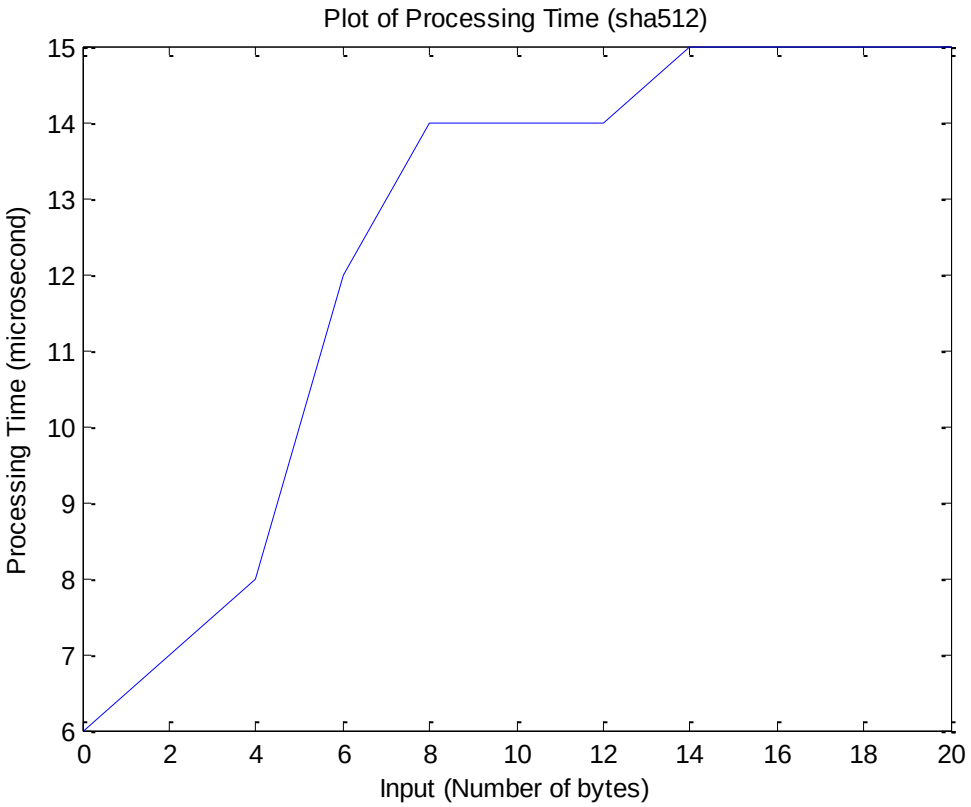

Figure 33. Plot of Processing Time (sha512).

After grasping the speed of both of the algorithms, it is suitable to compare them in one plane. **Figure 34** compares MD5 and sha512 algorithms with respect to the speed. It can be seen that MD5 is faster in comparison to the sha512 algorithm. On the other hand, sha512 is slow for the large input data bytes. Both of the algorithms have good results depending on the situation of the data processing. MD5 has been used in many Linux based operating systems while sha512 is relatively new and is also used in new Linux based operating systems.

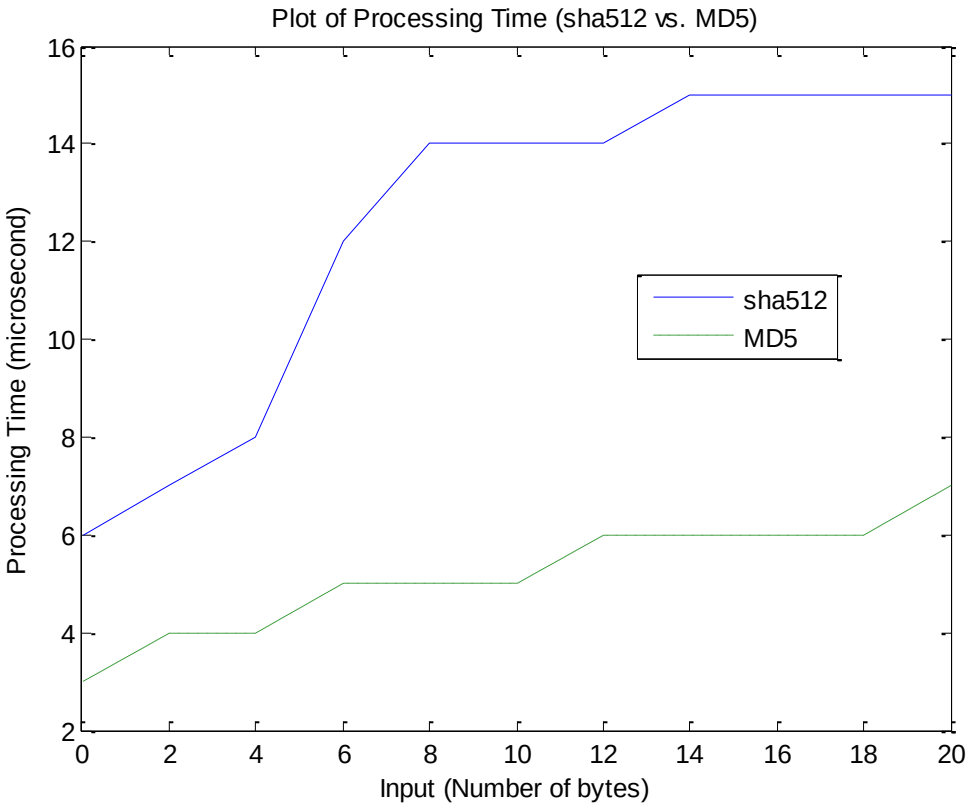

Figure 34. Processing Time (sha512 vs. MD5).

5.4 Cloud Computing Simulation

It is necessary to learn the behavior of cloud computing through simulation before actually deploying it. For this purpose, CloudSim is used. CloudSim provides the generalized and extensible simulation framework that enables experimentations of the Cloud Computing technology including its application services and infrastructure. It analyzes the various design issues of cloud computing and allows the user to be less concerned about the low level details related to the cloud computing infrastructure.

In this experiment, the submission of the request time and the finishing time has been scrutinized. First the simulation run on the Java SDK under Eclipse framework and then a Graphical User Interface is used to plot the graph of the simulation values (see **Figure 36**).

A hypothetical scenario is established to analyze the behavior of cloud computing. It consists of one Data center, one Cloudlet, one Host and one Virtual Machine. The specification of the Host and the Virtual Machine is defined in **Table 2**. The Cloudlet stores the list of the ID of the Virtual Machine despite of all the information encapsulated in the cloud and determines what needs to be done in the cloud. When a request is initiated, the Cloudlet decides how to schedule the resources to provide the service requested with the help of the Virtual Machine. The block diagram in **Figure 35** explains the scenario in detail.

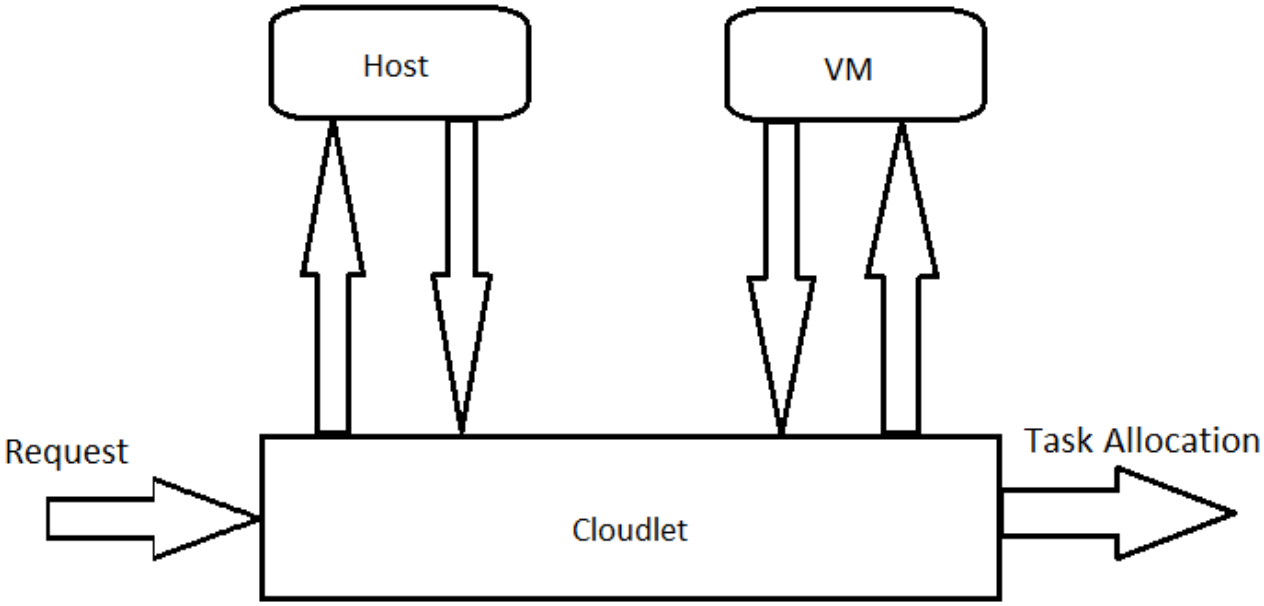

Figure 35. Hypothetical Scenario of the Data Center in the Cloud.

The setup values defined in the simulator are as follows:

Content	**Values**	**Content**	**Values**	**Content**	**Values**
Repeated Simulation Runs	10	Cloudlet created in each request	2	Host (H)	1
(H) RAM	16384Mb	(H) Storage	1000000 Mb	(H) Bandwidth	100000
Architecture	x86	OS	Linux	VM	Xen
VM Storage	10000 Mb	VM RAM	512 Mb	VM Bandwidth	1000
Number of CPU	1	Number of VM	1	Cost per Bandwidth	0.1 €
Cost per Host	3 €	Cost per Memory	0.05 €	Cost per Storage	0.1 €

Table 2. Input values to the Simulator.

The output obtained from Java is given below:

```
Starting CloudSim Simulation for one Datacenter and one Virtual Machine...
Initialising...
Starting CloudSim version 3.0
Datacenter is starting...
Broker is starting...
Entities started.
0.0: Broker: Cloud Resource List received with 1 resource(s)
0.0: Broker: Trying to Create VM #0 in Datacenter
0.1: Broker: VM #0 has been created in Datacenter #2, Host #0
0.1: Broker: Sending cloudlet 0 to VM #0
400.1: Broker: Cloudlet 0 received
400.1: Broker: All Cloudlets executed. Finishing...
400.1: Broker: Destroying VM #0
Broker is shutting down...
Simulation: No more future events
CloudInformationService: Notify all CloudSim entities for shutting down.
Datacenter is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.

========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time
    0        SUCCESS        2               0        400        0,1        400,1
Cloud Simulation Finished!
```

The results of the simulation are plotted in **Figure 36**. The unit of Time is given in milliseconds.

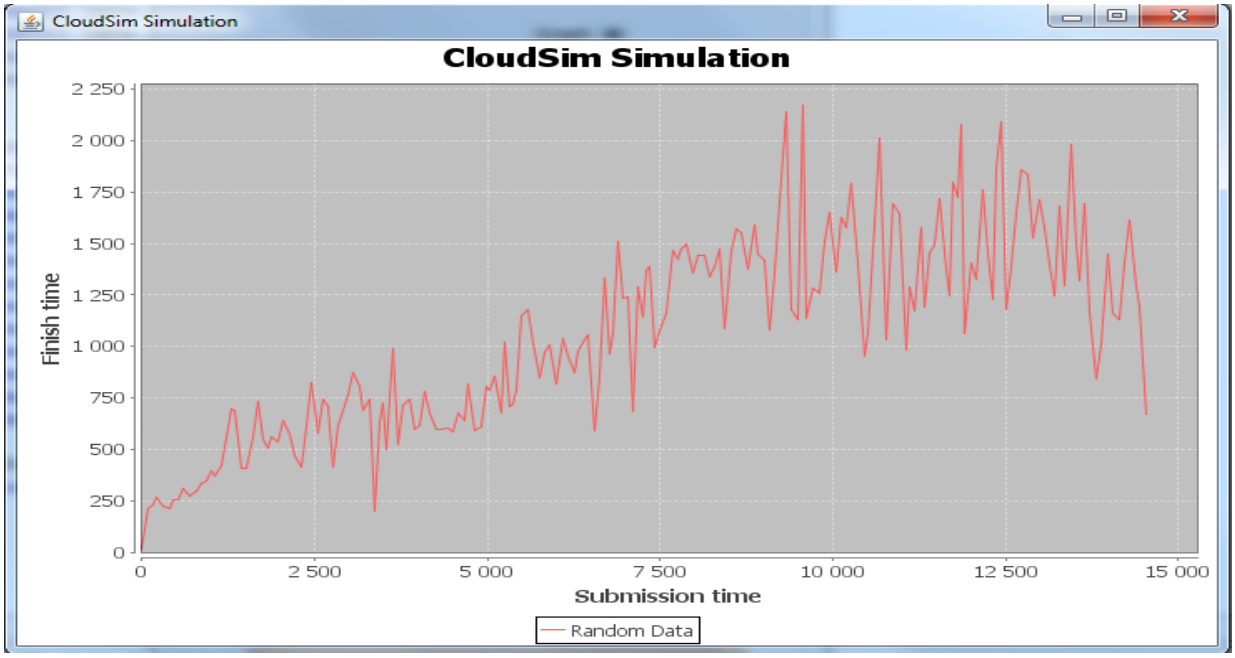

Figure 36. Plot between the submission time and finish time in the Cloud.

6. CONCLUSIONS AND FUTURE WORK

In this thesis, an integration of the Wireless Sensor Network and the cloud computing has been performed which is capable of getting the data from the sensor node (for example. Temperature, Light and battery information) and saving them into a database. A developed website displays the data stored in the database. The goal of the thesis is to provide the information received from the sensor node on the cloud so that it can be accessible from anywhere in the world. This project is applicable to small areas where measurements from the environment needs to be collected over a small wireless sensor network and where the collected data should be accessible through the Internet, which is an economical and effective way. These characteristics make this research topic in IT interested.

This thesis includes hardware as well as a software implementation for which the Wireless Sensor Network technology has been analyzed and a number of programming languages have been used. Critical aspects like security, distance estimation and transmission power of Wireless Sensor Network are investigated. The Sensor nodes are programmed according to the required conditions. One of the important parts of this project is the Graphical User Interface (GUI) which is programmed in Java to store the data received from Wireless Sensor Network in a database. A web-server has been installed and configured to host the website which shows the information of the environment received from the sensor network. PHP and MySQL have been used to build the website that reads the data from the database and displays them.

A good understanding of Wireless Sensor Networks as well as programming languages such as Java, embedded C, PHP and MySQL is required to build this project. It also requires knowledge about the Linux OS, Web Server configuration, system security and Cloud computing.

After having the grasp on the performance evaluation of Wireless Sensor Network, it is enlighten that the range of the WSN is limited for the communication between the sensor nodes which can be improved by utilizing the higher ranged transceivers depending on the scenario. The GUI application used in this project can be enhanced to a GUI applet (web application) in the future. The cloud server can be enhanced to provide the cloud service for multiple users in a virtualized environment.

REFERENCE

Slijepcevic, Sasha, Ranjit Iyer & Michael Panossian (2005). A survey of Wireless Sensor Networks.

Virtual-lab (2012). Simulating a Wireless Sensor Network. Available from the Internet: <URL: http://virtual-labs.ac.in/cse28/ant/ant/8/theory/>.

Jangra, Ajay, Swati, Richa & Priyanka (2010). Wireless Sensor Network (WSN). Architectural Design issues and Challenges.

Baghyalakshmi, D, Jemimah Ebenezer & S.A.V Satyamurty (2011).WSN based Temperature Monitoring for High . Performance Computing Cluster .

WSN-advanced-computer-science (2010). Special Issue "Wireless Sensor Network and Its Application in Advanced Computer Science". ISSN 1424-8220.

TSU (2008). Energy efficient Wireless Sensor Network. Texas Southern University. Available from the Internet: <URL: http://cs.tsu.edu/CREST/Research.aspx>.

Liu, Ling & Bugra Gedik. In Network Aggregation with Grouped Queries in multi-hop Wireless Sensor Networks. Available from the Internet: <URL: http://www.cc.gatech.edu/projects/disl/specialProjects/search.htm>.

Hac, Anna (2004). *Wireless Sensor Network Design*. ISBN: 0-470-86736-1

Ott, Alan (2012). Wireless Networking with IEEE 802.15.4 and 6LoWPAN . Embedded Linux Conference – Europe . IEEE.

Dynamic C (2008). An Introduction to Zig Bee. Available from the Internet: <URL: http://www.rabbit.com>.

Fu, Chunyan (2008). TCP/UDP Basics (Ericsson).

Modares, Hero, Rosli salleh & Amirhossein Moravejosharieh (2011). Overview of Security Issues in Wireless Sensor Networks.

Dunkels, Adam (2008). Contiki Crash Course.

Dunkels, Adam, Bjorn Gronvall & Thiemo Voigt (2004). Contiki - a lightweight and flexible operating system for tiny networked sensors.

Dunkels, Adam, Oliver Schmidt, Thiemo voig &, Muneeb Ali (2006). Protothread: Simplifying event-driven programming of memory constrained embedded systems .

Guan, Huiwei, Horace H. S. Ip & Yanchun Zhang (1998). Java-based approach for accessing database on the internet and JDBC-ODBC implementation.

Ege, Raimund K. (2000). Reading Large Volumes of Java Objects from Database .

Raj, Minu (2012). JDBC Architecture. Available from the Internet: <URL: http://minurajkattakada.blogspot.fi/2012/02/v-behaviorurldefaultvmlo.html>.

Hill, Benjamin Mako, Matthew Helmke & Corey Burger (2010). *The official ubuntu book.* ISBN-13: 978-0-13-708130-1

Bauer, Eric and Randee Adams (2012). *Reliability and availability of cloud computing.* ISBN 978-1-118-17701-3.

Brodkin, Jon(2010). Software as a Service. Available from the Internet: <URL: http://www.networkworld.com/news/2010/092710-software-as-service-security.html>.

Gibson, Joel, Robin Rondeau, Darren Eveleigh and Qing Tan (2012). Benefits and Challenges of Three Cloud Computing Service Models.

Habib, Shiekh M, Sascha Hauke, Sebastian Ries, Max Muhlahauser (2012). Trust as a facilitator in cloud computing: a survey journal of cloud computing: advances, systems and application.

Lenk, Alexander, Markus Klem, Jens Nimis, Stefan Tai and Thomas Sandholm (2009). What's inside the cloud? An architecture map pf the cloud landscape.

Jia, Minrui (2011). Cloud security of cloud computing application.

Jadeja, Yashpalsinh and Kirit Modi (2009). Cloud computing- concept, architecture and challenges.

Tianfield, Huaglory (2010). Security issues in cloud computing.

Thomas B, Winans And Seely Brown (2009). Demystifying Clouds. Exploring Cloud and Service Grid Architectures.

Zhou, Minqi, Rong Zhang, Dadan Zeng and Weining Qian (2010). Service in the cloud computing era: a survey.

Bell Labs (Rauscher 2006). Eight Ingredients of Communications Infrastructure: A Systematic and Comprehensive Framework for Enhancing Network Reliability and Security.

Wikipedia (2013 a). Cloud Computing. Available from the Internet:
<URL: http://en.wikipedia.org/wiki/Cloud_computing>.

Wikipedia (2013 b). Sensor Node. Available from the Internet:
<URL: http://en.wikipedia.org/wiki/Sensor_node>.

Wikipedia (2013 c). PHP. Available from the Internet:
<URL: http://en.wikipedia.org/wiki/PHP>.

Wikipedia (2013 d). SQL. Available from the Internet:
<URL: http://en.wikipedia.org/wiki/MySQL>.

Wikipedia (2013 e). Apache. Available from the Internet:
<URL: http://en.wikipedia.org/wiki/LAMP_%28software_bundle%29>.