INDOOR PERFORMANCE OF WIRELESS SENSOR NETWORK

HASRI BIN HAMDAN

A thesis submitted in partial
Fulfillment of the requirement for the award of the
Degree of Master of Electrical Communication Engineering

Faculty of Electrical and Electronic Engineering
Universiti Tun Hussein Onn Malaysia

JULY 2012

ABSTRACT

Wireless sensor networks have the potential to become significant subsystems of engineering applications where every each node functions as transmitter, receiver, router and data sink. It is necessary to understand the dynamic behaviour of these systems in simulation environments. It is critical to develop simulation platforms that are useful which can be used to explore both networking and wireless sensor networks issues. A discrete-event simulation is a trusted platform for modeling and simulating a variety of systems. This project emphasize on using new simulator for wireless sensor networks that is based on the discrete event simulation framework called Objective Modular Network Test bed in C++ version 4.1 (OMNeT++4.1) Simulator. This simulator is used to test the performance of sensor nodes within the networking in wireless communication networks based on indoor scenario. The test performances are focussed on aspects such as the time delay and packet utilization of the particular approach. The analysis approach is done through simulation software by the following metrics: packet frames delivery, packet loss and time delay experience within the system.

ABSTRAK

Rangkaian pengesan tanpa wayar mempunyai potensi untuk menjadi subsistem penting dalam aplikasi kejuruteraan di mana setiap nod boleh berfungsi sebagai pemancar, penerima, router dan sink data. Ia adalah perlu untuk memahami tingkah laku dinamik sistem-sistem ini dalam persekitaran suatu simulasi. Ia adalah sangat penting dalam membangunkan sebuah platform simulasi yang berguna untuk digunakan dalam meneroka isu-isu rangkaian dan rangkaian pengesan tanpa wayar. Penyelakuan diskret-acara adalah satu platform yang dipercayai untuk pemodelan dan simulasi pelbagai sistem. Projek ini menekankan penggunaan simulator baru ini bagi suatu rangkaian pengesan tanpa wayar berdasarkan rangka kerja simulasi peristiwa diskret yang dipanggil Simulator Ujian Objektif Rangkaian Modular Katil dalam C + + versi 4.1 (OMNeT + 4,1). Simulator ini digunakan untuk menguji prestasi nod pengesan dalam sesuatu rangkaian komunikasi tanpa wayar berdasarkan senario yang tertutup. Penilaian prestasi ujian tertumpu kepada aspek-aspek seperti penangguhan masa dan penghantaran paket berdasarkan pendekatan tertentu. Pendekatan analisis dilakukan melalui perisian simulasi melalui metrik berikut: penghantaran rangka paket, kehilangan paket dan penangguhan masa berlandaskan dalam sistem.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

WSN	-	Wireless Sensor Network
IEEE	-	Institute of Electrical and Electronic Engineers
OMNeT++	-	Objective Modular Network Test-bed in C++
RSSI	-	Received Signal Strength Indication
CPU	-	Central Processing Unit
OPNET	-	Optimized Network Engineering Tools
TCP/IP	-	Transmission Control Protocol/Internet Protocol
GUI	-	Graphical User Interface
NED	-	Network Description
MiXiM	-	Mix Simulator
MANET	-	Mobile Ad Hoc Network
PAN	-	Personal Area Network
WPAN	-	Wireless Personal Area Network
LR – WPAN	-	Low Rate Wireless Personal Area Network
ZDO	-	Zigbee Device Object
CAP	-	Contention Access Period
CFP	-	Contention Free Period
CSMA – CA	-	Carrier Sense Multiple Access – Collision Avoidance
MAC	-	Medium Access Control
FFD	-	Full Function Device
RFD	-	Reeduce Function Ddevice
LOS	-	Line Of Sight
OSI	-	Open System Interconnection
PL	-	Path Loss
α	-	Path Loss Exponent
P_{RX}	-	Received Power
d	-	Length or Distance Path
h	-	Antenna Height

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Background and History

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

In general, wireless sensor networks have made a lot of progress recently and have been widely discussed in many applications. According to J.Kenyeres et al (2010) it is expected that this technology will play an important role in improving the quality of the living environment through the creation of so called sensing environments. However, there is a gap in knowledge about WSN to help at least not to broaden this gap, but it is important that some scientific and educational research should be done in this area and that young generation should gain opportunity to study this technology.

A wireless sensor network consists of spatially distributed autonomus sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants. The development of wireless sensor networks was motivated by the military applications such as battlefield surveillance. Nowadays, it is also used in many industrial and civilian application areas, including industrial process monitoring and control, machine health monitoring, environment and habitat monitoring, healthcare applications, home automation, and traffic control.

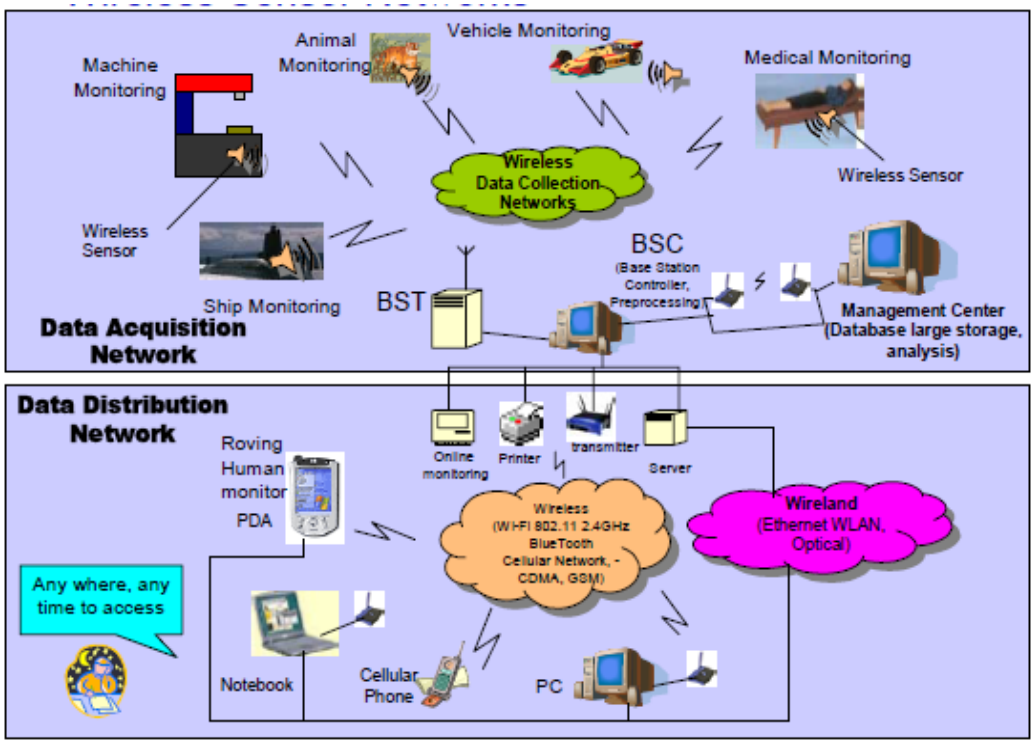

Figure 1.1 : A Wireless Sensor Network. (F.L.Lewis, 2004)

Feng Chen et al (2010) indicate that in wireless sensor network deployment techniques experiment, the important parameters such as connectivity of nodes, cost effective, energy efficiency and lifetime. Experimentation might be too expensive for such setups or infeasible due to physical limitations. Thus, in most cases performance evaluation is based on simulation models. For this study, the simulation experiment on the coexistence between wireless technologies based on IEEE 802.15.4 and IEEE 802.11b standards can be considered using OMNeT++ software. OMNeT++ software is suitable to implement simulation experiment and to evaluate the performance of the WSN network within indoor environment.

1.2 Problem Statement

In the last decade, significant advances have been achieved in the domain of Wireless Sensor Networks (WSN). In the recent years, Wireless Sensor Networks probably plays a crucial role in creation of ubiquitous intelligent sensing environment. According to J.Kenyeres et al (2010), WSN are suitable for great variety of environments and conditions, what broadens the scale of their applications. Furthermore, simulation is frequently used to evaluate the performance of networking algorithms and techniques in wireless communication networks. Feng Chen et al (2010) indicate that however, performance aspects such as the transmission delay, the channel utilization, or the throughput provide only limited information about the feasibility of the particular approach. Some of the most challenging issues that have been studied are the medium access, routing strategies, clustering schemes, and application layer dynamics. All these approaches contribute to enable designers to develop and to deploy applications under various environmental conditions.

The idea is to provide a broad range of design variants that can be chosen and combined in order to provide the optimal behaviour of the wireless sensor network. In certain cases, monitoring and automatic control of building environment is a crucial application of WSN in which maximizing network lifetime. It shows that transitional region is particular concern its accommodates high variance unreliable links due to the inside building environment could be the obstacles such as concrete or brick walls, partitions, office furniture and other items as additional absorption term to the path loss according to C. Mallanda (2005). Most of the approaches are targeted to improve the performance of the wireless communication with respect to the quality of service. Therefore, all the individual algorithms and techniques have been analyzed with regard to their performance, example, the speed of adaptation to environmental changes and the end-to-end nodes performance. Furthermore, most of the aim in wireless communication networks and especially for sensor networks is to reduce the energy consumption.

1.3 Objectives of Project

The main objectives of this project are:

i. To design a simulation process for wireless sensor network in an indoor scenario using OMNeT++ 4.1
ii. To test the performance of packet delivery among the nodes within indoor scenario using OMNeT++4.1

1.4 Scope of Project

The scopes of this project is to evaluate wireless sensor network routing through research and simulation process. An open source software called OMNeT++ version 4.1 will be used in this project to run simulation process according to certain parameters that acquired according to indoor scenario. The parameters such as numbers on nodes, transmitting power used by the nodes and distances between nodes. Throughout the simulation process, the performances of packet delivery among the nodes within the network will be analyzed based on the results.

1.5 Thesis Layout

The thesis layout is organized as follows:

Chapter 1: This chapter explain on the introduction to wireless sensor network (WSN). It is also consists of background, problem statement, objectives of project and scope of project.

Chapter 2: In this chapter, it discussed on literature reviews of other research or previous studies which conclude theoritical and results. It is also to clarified, justified and compared between results based on related research.

Chapter 3: In order to achieve the goal of this project, this chapter will expalin on the methodology of the project. It showed the steps or protocols used in completion of this project. The simulation process and parameters are also explained and discussed in this chapter.

Chapter 4: The results of this studies are presented and compared within this chapter. Analysis and simulation output and comparison between results such as graphs are explained in this chapter.

Chapter 5: Lastly, this chapter will summarizes all the results and concluded the conclusions of the project studies including the recommendation for future works.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

The objective of this chapter is to survey on previous studies that had been carried out among other researchers in order to gain more information related to the project. It is critically important to review the existing research on wireless sensor network routing protocols within indoor scenario. By critically reviewing the following previous thesis, journals, articles and reports, substantially it provides great significance knowledge and information to this project.

2.2 Related Works

In this section, related works on the routing protocol for WSN that improve the energy consumption are studied and critically analyzed.

J. Kenyeres et al. (2010) focused on the monitoring of functionality and reliability as well as on the influence on varying parameters of wireless sensor network placed in the indoor environment. The experiment clearly revealed the relationship between data rate and reponsibility of the whole WSN network. In reducing error rate and increase number of packets for static WSN can be achieved by reducing topology, decrease the distance between nodes and increase packet sending interval.

Malka N. Halgamuge et al. (2009) investigate the link quality distribution to obtain full coverage of the signal strength within a single floor of a building environment. The results comfirmed where the transitional region in wireless sensor network since it accomodates high variance unreliable links. The reason is due to this transitional region

inside a building environment could be the obstacles including concrete walls, partitions, furnitures and other items affect as additional absorption term to path loss.

Yunchun Zhang et al. (2009) presented the existing localization algorithm in Wireless Sensor Network (WSN) which can be divided into two categories: range based and range free. Most of the range based localization algorithms proposed made use of the Received Signal Strength Indication (RSSI) to make an estimation of the distance between transmitter and receiver. Throughout the experiment it shows there is no relationship between RSSI and distance in indoor situations. Only the outdoor shows that RSSI is closely related with distance, direction of antenna, the height of nodes above the ground and obstructions.

D. J. Dechene examined currently proposed clustering algorithms for Wireless Sensor Networks (WSN) which discussed the operations of the algorithms, as well as comparisons on the performance between the various schemes. Optimal clustering in terms of energy efficiency should eliminate all overhead associated not only with the cluster-head selection process, but also with node association to their respective cluster-heads. Sensor network reliability is currently addressed in various algorithms by utilizing re-clustering that occurs at various time intervals; however inefficient energy and limits the time available within a network for data transmission and sensing tasks. Further improvements in reliability should examine possible modifications to the re-clustering mechanisms following the initial cluster-head selection. Other mechanisms such as the ability of nodes to maintain membership in auxiliary clusters can reinforce the current state of sensor network reliability.

Feng Chen et al. (2010) introduced a generic energy model developed for the simulation framework OMNeT++. The sensor was designed based on the IEEE802.14.5 architecture by using a simple CPU model to estimate the energy consumption for computationally intensive operations. The aspect discuss in this paper is more on the sensor design and not taking into account the routing algorithm. The model allows to accurately evaluate the energy performance in terms of energy consumption or network lifetime of the wireless sensor network.

2.3 Wireless Sensor Network

Wireless Sensor Networks (WSN) comprises of numerous tiny sensors that are deployed in spatially distributed terrain. These sensors are endowed with small amount of computing and communication capability and can be deployed in ways that wired sensor systems couldn't be deployed. Yunchun Zhang et al. (2009) indicates that despite the prolific conceptualization of sensor networks as being useful for large-scale military applications which originally motivated by military applications such as battlefield surveillance, is a wireless network consisting of spatially distributed autonomous devices using sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants, at different locations.

Wireless sensor networks have the potential to become significant subsystems of engineering applications. Before relegating important and safety-critical tasks to such subsystems, it is necessary to understand the dynamic behavior of these subsystems in simulation environments. C Mallanda et al. (2005) urge that it is an urgent need to develop simulation platforms that are useful to explore both the networking issues and the distributed computing aspects of wireless sensor networks.

According to J. Kenyeres et al. (2010) of the experiments aim was to estimate how parameters such as number of packet errors and retransmissions deteriorate real WSN functionality. The experiment with WSN was realized in indoor area, which resulted in a variety of different conditions influencing functionality of sensor nodes. This area offers lot of different spaces ranging from long, open sight hallways, stairways, lifts and many others. This surrounding environment at the main hallway consists of many different materials, like concrete, steel, glass and so on, what causes heterogeneous influences on WSN.

WSN signal in this area might be interfaced with signals from other sources, for example from 802.11b communication. Main goal of this experiment was to create WSN network covering almost whole area of hallway and also to test its functionality with various parameters set up. Discrete-event simulation is a trusted platform for modeling and simulating a variety of systems. Results can be obtained from a new simulator for wireless sensor networks networks that is based on the discrete event simulation framework called OMNeT++.

Work is underway to develop a simulation platform that allows developers and researchers to investigate topological, phenomenological, networking, robustness and

scaling issues related to wireless sensor networks. Such simulation studies must explore the effects of scale, density, node-level architecture, energy efficiency, communication architecture, failure modes at node and communication media levels, system architecture, algorithms, protocols and configuration among other issues. C Mallanda et al. (2005) told that unlike traditional computer systems, it is not sufficient to simulate the behavior of the sensor network in isolation because of the tight and ubiquitous coupling between the sensor network and its application.

2.4 Objective Modular Network Test-bed in C++ (OMNeT++)

Thru simulation, there are many types of current available simulators such as ns2, SensorSim, OPNET, J-Sim, GlomoSim and etc. all these simulators have certain function that provides support or platform for simulating TCP/IP, routing protocols, energy models, sensor channels and etc. In this project, the selected simulator used is Objective Modular Network Test-bed in C++ (OMNeT++).

2.4.1 Background And History

OMNeT++ started with a programming assignment at the Technical University of Budapest (Hungary), to which two students applied. One of them, András Varga, still is the maintainer of this open source simulation package. During the years several people contributed to OMNeT++, among which several students from the Technical University of Delft (The Netherlands) and Budapest. Milestones in the development are the first public release in 1997 and the added animation functionality in 1998, which made the package even more usable for education.

In 2000 several people at the University of Karlsruhe created the TCP model for OMNeT++. This version included several bug fixes and important changes and additions to the manual. The website also had a major update. A beta version of version 3.0 is now also available. All simulations in this report were done using OMNeT++ version 4.1. It is also offers an Eclipse-based IDE, a graphical runtime environment, and a host of other tools. There are extensions for real-time simulation, network emulation, alternative programming languages (Java, C#), database integration, SystemC integration, and several other functions. The latest extension version is OMNeT++4.2.

2.4.2 What Is OMNeT++

OMNeT++ is an extensible, modular, component-based C++ simulation library and framework, primarily for building network simulators. OMNeT++ is also an object oriented discrete event simulation environment focusing on the simulation of communication networks. It provides component architecture for models programmed in C++ with GUI support. Its primary application area is the simulation of communication networks, but because of its generic and flexible architecture, is successfully used in other areas like the simulation of complex IT systems, queueing networks or hardware architectures as well.

OMNeT++ provides a component architecture for models. Components *(modules)* are programmed in C++, then assembled into larger components and models using a high-level language *(NED)*. Reusability of models comes for free. OMNeT++ has extensive GUI support, and due to its modular architecture, the simulation kernel (and models) can be embedded easily into any applications. Although OMNeT++ is not a network simulator itself, it is currently gaining widespread popularity as a network simulation platform in the scientific community as well as in industrial settings, and building up a large user community.

2.4.3 Applications of OMNeT++

A *C++ class library* which consists of the simulation kernel and utility classes (for random number generation, statistics collection, topology discovery etc) This will be use to create simulation components (*simple modules* and channels); infrastructure to assemble simulations from these components and configuration (*NED language, ini files*); runtime user interfaces or *environments* for simulations (*Tkenv, Cmdenv*); an Eclipse-based simulation IDE for designing, running and evaluating simulations; extension interfaces for real-time simulation, emulation, MRIP, parallel distributed simulation, database connectivity and so on.

OMNeT++ provides the fundamental machinery and tools in writting simulations, but it does not provide any components specifically for computer network simulations, queueing network simulations, system architecture simulations or any other area. Instead, these application areas are supported by various simulation models and frameworks of an

open source such as INET/INETMANET, MiXiM or Castalia. These models are developed completely independent of OMNeT++, and follow their own release cycles.

2.4.4 Modelling Concepts

OMNeT++ provides efficient tools for the user to describe the structure of the actual system. Some of the main features are:

- hierarchically nested modules
- modules are instances of module types
- modules communicate with messages through channels
- flexible module parameters
- topology description language

2.4.4.1 Hierarchical Modules

An OMNeT++ model consists of hierarchically nested modules, which communicate by passing messages to each another. OMNeT++ models are often referred to as networks. The top level module is the system module. The system module contains submodules, which can also contain submodules themselves (Fig. 2.1). The depth of module nesting is not limited; this allows the user to reflect the logical structure of the actual system in the model structure.

Model structure is described in OMNeT++'s NED language.

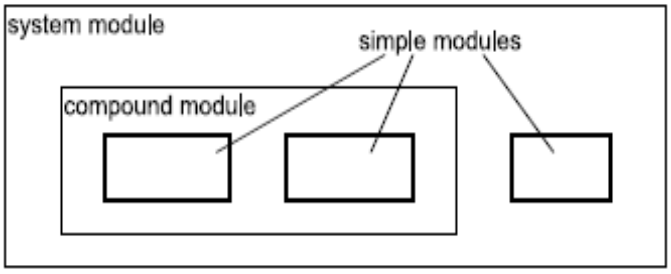

Figure 2.1: Simple and compound modules

Simple modules are the active components in the model. Simple modules are programmed in C++, using the OMNeT++ class library. The following sections contain a short introduction to discrete event simulation in general, explain on how its concepts are

implemented in OMNeT++, and give an overview and practical advice on how to design and code simple modules.

Modules that contain submodules are termed compound modules, as opposed simple modules which areat the lowest level of the module hierarchy. Simple modules contain the algorithms in the model. The user implements the simple modules in C++, using the OMNeT++ simulation class library.

In OMNeT++, events occur inside simple modules. Simple modules encapsulate C++ code that generates an events and reacts to events, in other words, implements the behaviour of the model. The user creates simple module types by sub classing the cSimpleModule class, which is part of the OMNeT++ class library. cSimpleModule, just as cCompoundModule, is derived from a common base class, cModule. cSimpleModule, although packed with simulation-related functionality, doesn't do anything useful by itself . It needs to redefine some virtual member functions to make it do some useful work. These member functions are the following:

- void initialize()
- void handleMessage(cMessage *msg)
- void activity()
- void finish()

In the initialization step, OMNeT++ builds the network: it creates the necessary simple and compound modules and connects them according to the NED definitions. OMNeT++ also calls the initialize() functions of all modules. The finish() functions are called when the simulation terminates successfully. The most typical use of finish() is the recording of statistics collected during simulation.

2.4.4.2 Module Types

Both simple and compound modules are instances of module types. While describing the model, the user defines module types; instances of these module types serve as components for more complex module types. Finally, the user creates the system module as an instance of a previously defined module type; all modules of the network are instantiated as sub modules and sub-sub modules of the system module. When a module type is used as a building block, there is no distinction whether it is a simple or a compound module. This allows the user to split a simple module into several simple modules embedded into a compound module, or vice versa, aggregate the functionality of

a compound module into a single simple module, without affecting existing users of the module type. Module types can be stored in files separately from the place of their actual usage. This means that the user can group existing module types and create component libraries.

2.4.5 Components of OMNeT++

There are several types of components used in OMNeT++'s simulation. The following main components are as follows:

- simulation kernel library
- compiler for the NED topology description language
- OMNeT++ IDE based on the Eclipse platform
- GUI for simulation execution, links into simulation executable (Tkenv)
- command-line user interface for simulation execution (Cmdenv)
- utilities (makefile creation tool, etc.)
- documentation, sample simulations, etc.

2.4.6 Simulation with OMNeT++

This section provides insight into working with OMNeT++ in practice: Issues such as model files, compiling and running simulations are discussed.

An OMNeT++ model consists of the following parts:

- NED language topology description(s) (.ned files) which describe the module structure with parameters, gates etc. NED files can be written using any text editor or the GNED graphical editor.
- Message definitions (.msg files). Define various message types and add data fields to them. OMNeT++ will translate message definitions into full-fledged C++ classes.
- Simple modules sources. They are C++ files, with .h/.cc suffix.

The simulation system provides the following components:

- Simulation kernel. This contains the code that manages the simulation and the simulation class library. It is written in C++, compiled and put together to form a library (a file with .a or .lib extension)

- User interfaces. OMNeT++ user interfaces are used in simulation execution, to facilitate debugging, demonstration, or batch execution of simulations. There are several user interfaces, written in C++, compiled and put together into libraries (.a or .lib files).

Simulation programs are built from the above components. First, .msg files are translated into C++ code using the opp_msgc. program. Then all C++ sources are compiled, and linked with the simulation kernel and a user interface library to form a simulation executable. NED files can either be also translated into C++ (using nedtool) and linked in, or loaded dynamically in their original text forms when the simulation program starts.

In figure 2.2, figure 2.3 and figure 2.4 shows the window box related to the NED graphical editor, source code editor and graphical runtime environment.

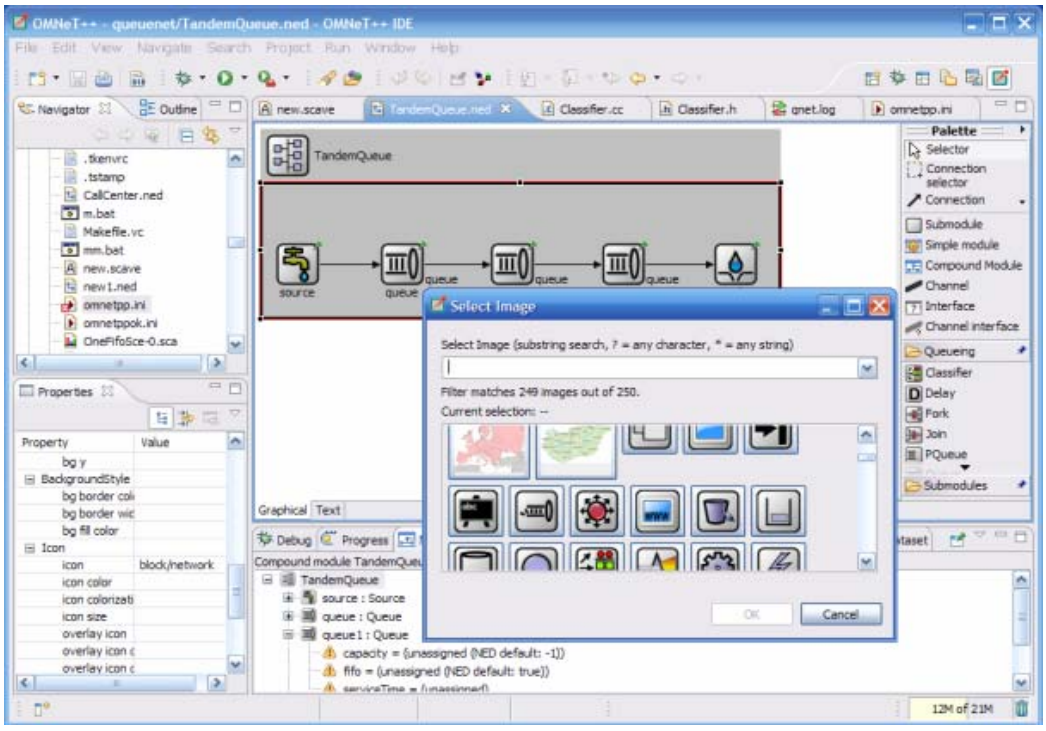

Figure 2.2: Graphical NED Editor

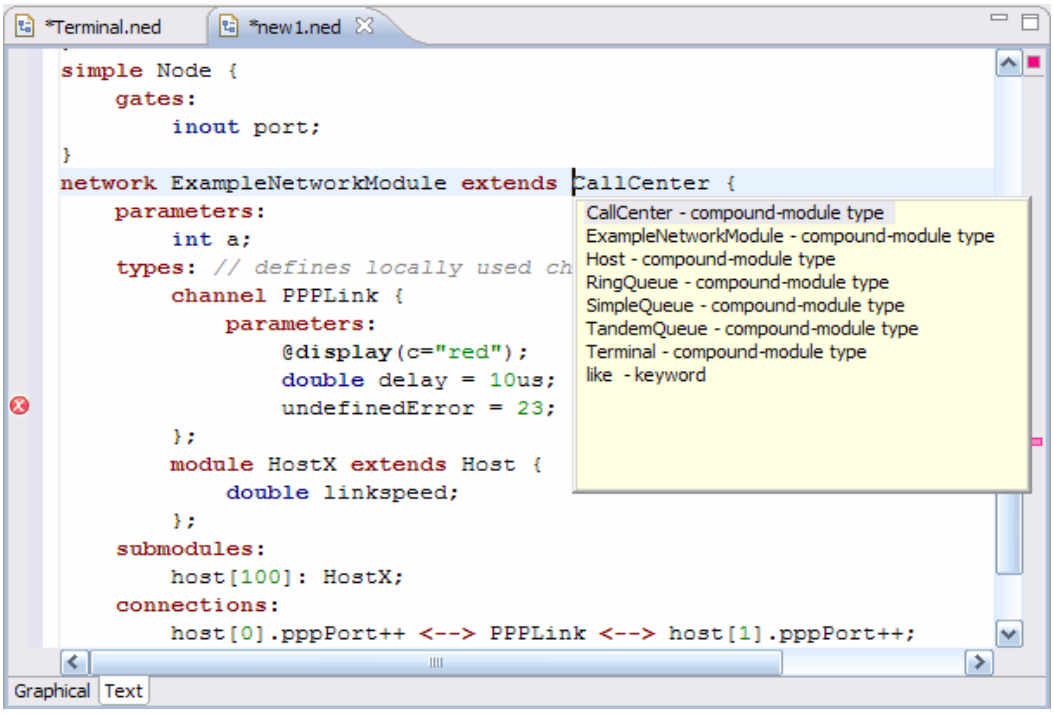

Figure 2.3: NED Source Editor

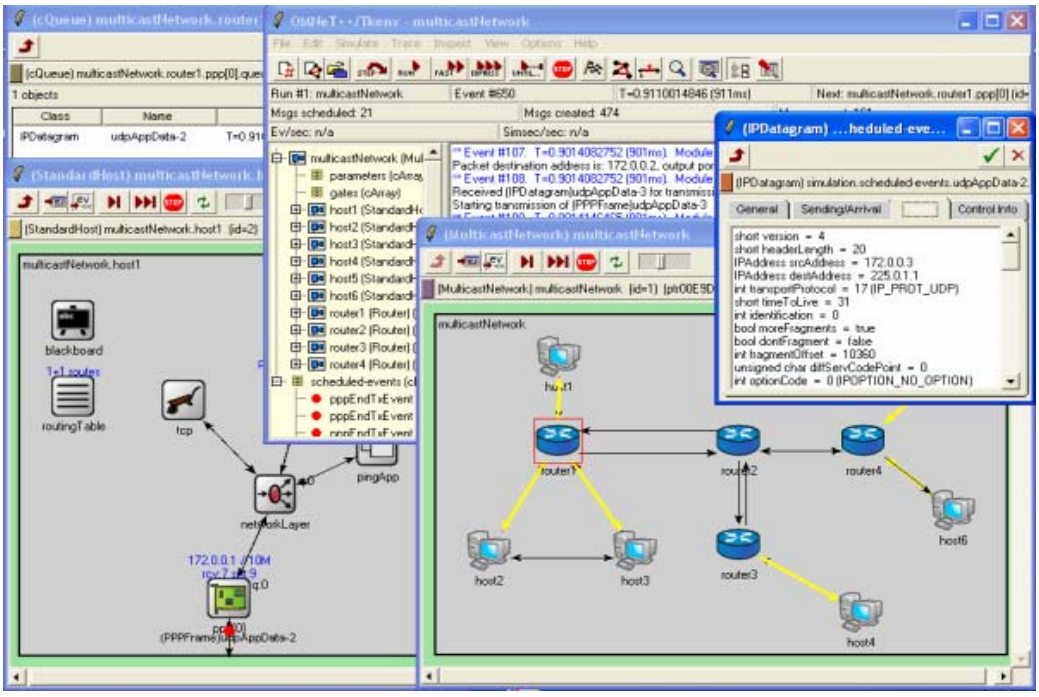

Figure 2.4: Graphical Runtime Environment

2.4.7 Running Simulation and Analyzing Results

The simulation executable is a standalone program, thus it can be run on other machines without OMNeT++ or the model files being present. When the program is started, it reads a configuration file (usually called omnetpp.ini). This file contains settings that control how the simulation is executed, values for model parameters, etc. The configuration file can also prescribe several simulation runs; in the simplest case, they will be executed by the simulation program one after another.

The output of the simulation is written into data files: output vector files, output scalar files, and possibly the user's own output files. OMNeT++ provides a GUI tool named Plove to view and plot the contents of output vector files. It is not expected that someone will process the result files using OMNeT++ alone: output files are text files in a format which can be read into math packages like Matlab or Octave, or imported into spreadsheets like Open Office Calc, Gnumeric or MS Excel. All these external programs provide rich functionality for statistical analysis and visualization, and it is outside the scope of OMNeT++ to duplicate their efforts. This manual briefly describes some data plotting programs and how to use them with OMNeT++.

2.4.8 The NED Language

The topology of a model is specified using the NED language. The NED language facilitates the modular description of a network. This means that a network description may consist of a number of component descriptions (channels, simple/compound module types). The channels, simple modules and compound modules of one network description can be reused in another network description. Files containing network descriptions generally have a .ned suffix. NED files can be loaded dynamically into simulation programs, or translated into C++ by the NED compiler and linked into the simulation executable.

The NED language, the network topology description language of OMNeT++ will be given using the extended BNF notation. Space, horizontal tab and new line characters counts as delimiters, so one or more of them is required between two elements of the description which would otherwise be unseparable. '//' (two slashes) may be used to write comments that last to the end of the line. The language only distinguishes between lower and upper case letters in names, but not in keywords.

In this description, the {xxx...} notation stands for one or more xxx's separated with spaces, tabs or new line characters, and {xxx,,,} stands for one or more xxx's, separated with a comma and (optionally) spaces, tabs or new line characters. For ease of reading, in some cases we use textual definitions. The network description symbol is the sentence symbol of the grammar.

2.4.9 Components Of A NED Description

A NED description consist of the following components, in arbitrary number or order:

- import directives
- channel definitions
- simple and compound module definitions
- network definitions

2.4.10 User Interfaces

OMNeT++ simulations can be run under different user interfaces. Currently, there are two types of user interfaces were supported:

- Tkenv: Tcl/Tk- which is based graphical, windowing user interface
- Cmdenv: a command-line user interface for batch execution

Typically test and debug are simulation which is under Tkenv, which run the actual simulation experiments from the command line or shell script, using Cmdenv. Tkenv is well suited for the use of educational or any demonstration purposes. Both Tkenv and Cmdenv are provided in the form of a library, and by choosing between them by linking one or the other into the simulation executable. Both user interfaces are supported on Windows or Unix platforms.

Common functionality in Tkenv and Cmdenv has been collected and placed into the Envir library, which can be thought of as the "common base class" for the two user interfaces. The user interface is separated from the simulation kernel, and the two parts interact through a well defined interface. This means that, if it is necessary needed, the user can write its own user interface or embed an OMNeT++ simulation into the application without any changes to models or the simulation library.

Configuration and input data for the simulation are described in a configuration file usually called omnetpp.ini. Some entries in this file apply to Tkenv or Cmdenv only,

other settings are in effect regardless of the user interface. Both user interfaces accept command-line arguments.

2.4.11 The Configuration File: omnetpp.ini

OMNeT++ is an open source simulator that can execute several simulations runs automatically, one after another without any distortion. If multiple runs are selected, option settings and parameter values can be given either individually for each run, or all together for the whole runs, depending in which of the section the option or parameter will appeared.

2.4.11.1 File Syntax

The *ini* file is a text file consisting of entries grouped into different sections. The order of the sections doesn't matter. Also, if there have two sections with the same name:
(e.g. [General] occurs twice in the file), they will be merged.

- Lines that start with "#" or ";" are comments, and will be ignored during processing.
- Long lines can be broken up using the backslash notation: if the last character of a line is "\", it will be merged with the next line.

The size of the ini file (the number of sections and entries) is not limited. Currently there is a 1024- character limit on the line length, which cannot be increased by breaking up the line using backslashes.

2.5 IEEE 802.15.4

An IEEE standard 802.15.4 intends to offer the fundamental lower network layers of a type of wireless personal area network (WPAN) which focuses on low-cost, low-speed ubiquitous communication between devices which in contrast with other, more end-user oriented approaches, such as Wireless Fidelity (WiFi). It emphasis on very low cost communication within a nearby devices with little to none underlying infrastructure, which intending more to exploit very low power consumption.

IEEE 802.15.4 is a standard which specifies the physical layer and media access control for low-rate wireless personal area networks (LR-WPANs). It is maintained by the IEEE 802.15 working group. It is the basis for the ZigBee specifications, which further extends the standard by developing the upper layers which are not defined in IEEE 802.15.4.

ZigBee is a specification for a suite of high level communication protocols using small, low-power digital radios based on an IEEE 802 standard for personal area networks. Applications include wireless light switches, electrical meters with in-home-displays, and other consumer and industrial equipment that requires short-range wireless transfer of data at relatively low rates. The technology defined by the ZigBee specification is intended to be simpler and less expensive than other WPANs, such as Bluetooth. ZigBee is targeted at radio-frequency (RF) applications that require a low data rate, long battery life, and secure networking. ZigBee has a defined rate of 250 kbit/s best suited for periodic or intermittent data or a single signal transmission from a sensor or input device. ZigBee based traffic management system have also been implemented.

2.5.1 Technical Overview

ZigBee is a low-cost, low-power, wireless mesh network standard. The low cost allows the technology to be widely deployed in wireless control and monitoring applications. Low power-usage allows longer life with smaller batteries. Mesh networking provides high reliability and more extensive range. ZigBee chip vendors typically sell integrated radios and microcontrollers with between 60 KB and 256 KB flash memory.

The ZigBee network layer natively supports both star and tree typical networks, and generic mesh networks. Every network must have one coordinator device, tasked with its creation, the control of its parameters and basic maintenance. Within star networks, the coordinator must be the central node. Both trees and meshes allows the use of ZigBee routers to extend communication at the network level.

Figure 2.5 shows the ZigBee protocol stack. The diagram is a compact representation of the complete ZigBee protocol stack, including IEEE 802.15.4-defined bottom layers. It builds upon the physical layer and medium access control defined in IEEE standard 802.15.4 for low-rate WPANs. The specification goes on to complete the standard by adding four main components: network layer, application layer, ZigBee device objects (ZDOs) and manufacturer-defined application objects which allow for customization and favor total integration.

Besides adding two high-level network layers to the underlying structure, the most significant improvement is the introduction of ZDOs. These are responsible for a number of tasks, which include keeping of device roles, management of requests to join a network, device discovery and security. ZigBee is not intended to support powerline networking but to interface with it at least for smart metering and smart appliance purposes. ZigBee nodes can go from sleep to active mode in 30 ms or less, the latency can be low and devices can be responsive, particularly compared to Bluetooth wake-up delays, which are typically around three seconds. Because ZigBee nodes can sleep most of the time, average power consumption can be low, resulting in long battery life.

The basic framework conceives a 10-meter communications range with a transfer rate of 250 kbit/s. Tradeoffs are possible to favor more radically embedded devices with even lower power requirements, through the definition of not one, but several physical layers. Lower transfer rates of 20 and 40 kbit/s were initially defined, with the 100 kbit/s rate being added in the current revision.

Even lower rates can be considered with the resulting effect on power consumption. As already mentioned, the main identifying feature of IEEE 802.15.4 among WPAN's is the importance of achieving extremely low manufacturing and operation costs and technological simplicity, without sacrificing flexibility or generality.

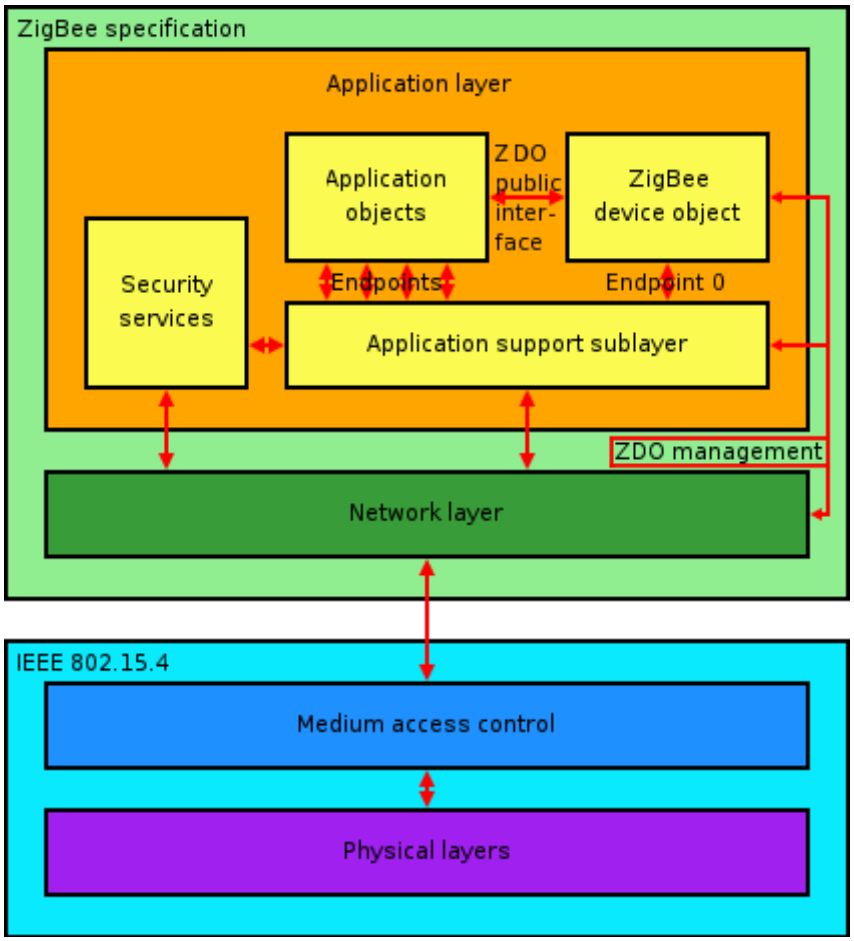

Figure 2.5: A Zigbee protocol stack

A Zigbee network can work in one of three ISM frequency bands and choose from a total of 27 channels. Two different types of devices are defined in an LR-WPAN, a full function device (FFD) and a reduced function device (RFD).

The first one is the FFD. An FFD can talk to any other device and serves as a PAN coordinator, a coordinator or a device. It can serve as the coordinator of a personal area network just as it may function as a common node. It implements a general model of communication which allows it to talk to any other device: it may also relay messages, in which case it is dubbed a coordinator (PAN coordinator when it is in charge of the whole network).

On the other hand there is RFD. An RFD can only talk to an FFD node. These are meant to be extremely simple devices with very modest resource and communication requirements; due to this, they can only communicate with FFD's and can never act as coordinators. The standard supports two network topologies, star and peer-to-peer. In the star network, the communication occurs only between devices and a single central controller, called the PAN coordinator, which manages the whole PAN. The peer-to-peer topology also has a PAN coordinator, however is differs from the star topology in that any devices can communicate with any other one as long as they are in range of one another.

To achieve better energy-efficiency, Zigbee can operate on beacon-enabled mode, for which a super frame structure is utilized. A super frame is bounded by periodically transmitted beacon frames, which allow nodes to associate with and synchronize to their coordinators. It consists of two parts, active and inactive period. An active portion is divided into 16 contiguous time slots that form three parts: the beacon, contention access period (CAP) and contention-free period (CFP). In CAP, all data transmission should follow a successful execution of the slotted CSMA-CA algorithm.

2.5.2 Description of IEEE 802.15.4 Model in OMNET++

The IEEE 802.15.4 model is developed in the MIXIM framework, which is an open-source communication networks simulation package for the OMNeT++ simulation environment and suited for simulations of wired, wireless and ad-hoc networks. The architecture of the 802.15.4 model is shown in Figure 2.6. There are three sub models, traffic, MAC and PHY, each of which is an independent module and inherited from the basic C++ class cSimpleModule in OMNeT++. The modules are connected with each other via gates and communicate among each other via messages.

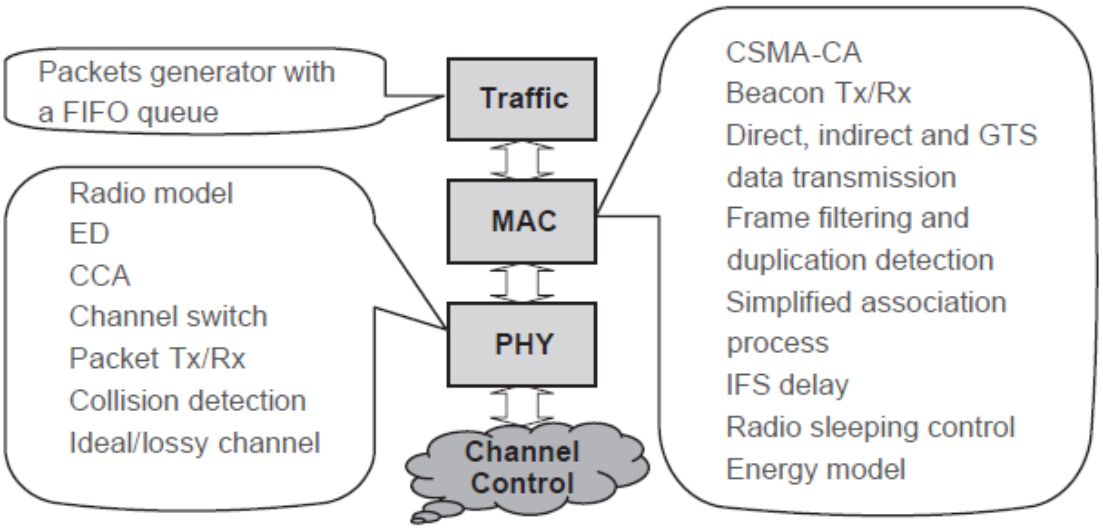

Figure 2.6: The structure and components of IEEE 802.15.4 model

2.5.3 Path Loss Indoor Propagation Model

Path loss or path attenuation is the reduction in power density (attenuation) of an electromagnetic wave as it propagates through space. Path loss is a major component in the analysis and design of the link budget of a telecommunication system. This term is commonly used in wireless communications and signal propagation. Path loss may be due to many effects, such as free-space loss, refraction, diffraction, reflection, aperture-mediumcoupling loss, and absorption. Path loss is also influenced by terrain contours, environment, propagation medium (dry or moist air), the distance between the transmitter and the receiver, and the height and location of antennas.

The model is applicable to indoor propagation modeling where log distance path loss model is formerly expressed as:

$$PL. = P_{Tx(dBm)} - P_{Rx(dBm)} \quad (2.1)$$
$$= PL_o + 10\,\alpha\, log_{10}\,(d/d_o) + X_g$$

where;

PL is the total path loss measured in Decibel (dB)

$P_{Tx(dBm)}=10\ log_{10}\ (P_{Tx}/1mW)$ is the transmitted power in dBm, where

P_{Tx} is the transmitted power in watt.

$P_{Rx(dBm)}=10\ log_{10}\ (P_{Rx}/1mW)$ is the received power in dBm, where

P_{Rx} is the received power in watt.

PL_o is the path loss at the reference distance d_0. Unit: Decibel (dB)

d is the length of the path.

d_o is the reference distance, usually 1 km (or 1 mile).

α is the path loss exponent.

X_g is a normal (or Gaussian) random variable with zero mean, reflecting the attenuation (in decibel) caused by flat fading.

When an electromagnetic wave propagates through space; there is the reduction in power density or attenuation of the wave, namely path loss, which is a major component in the channel modelling. According to Jing Lu et al. (2010), the simplest channel is the free space line of sight channel with no objects between the receiver and the transmitter or around the path between them. In this simple case, the transmitted signal attenuates since the energy is spread spherically around the transmitting antenna. For this line of sight (LOS) channel, the received power is given by:

$$PL\,(d)\,dB = \overline{PL}\,(d_o) + 10\alpha\,log\,(d/d_o)\,\sigma \qquad (2.2)$$

Some of the waves will reflect and reach the transmitter due to the presence of the ground. These reflected waves sometime have a phase shift of 180° and so may reduce the net received power. So, a simple two-ray approximation for path loss can be shown as below:

$$P_r = P_t\,(G_r G_t h_r^2 h_t^2 / d^4) \qquad (2.3)$$

Respectively, from the given formula, where h_r and h_t are the antenna heights of the transmitter and receiver. Note that there are three major differences from the previous formula. First, the antenna heights have effect. Second, the wavelength is absent and third the exponent on the distance is 4. In general, a common formula for path loss is:

$$P_r = P_t\,P_o\,(d_o/d)^{\alpha} \qquad (2.4)$$

Where P_o is the power at a distance d_o and α is the path loss exponent.

Theoretically, the power falls off in proportion to the square of the distance. In practice, the power falls off more quickly, typically 3rd or 4th power of distance. The

REFERENCES

Ali, Q.I., Abdulmaowjod, A., & Mohammed, H.M. (2011). Simulation and Performance Study of Wireless Sensor Network (WSN) Using MATLAB. *Irag J. Electrical and Electronic Engineering Vol. 7 No. 2,* 112-119.

Cetin, B. (2006). Opportunistic Relay Protocol for IEEE 802.11 WLANs. *Swedish Institute of Computer Science, Stockholm*, Sweden. March 2006.

Chen, F., Dietrich, I., German, R., & Dressler, F. (2010). An Energy Model for Simulation Studies of Wireless Sensor Network using OMNeT++. *Computer Networks and Communication Systems,* University or Erlangen, Germany.

Chen, F., & Dressler, F. (2007). A Simulation Model of IEEE 802.15.4 in OMNeT++. *Computer Networks and Communication Systems, University of Erlangen-Nuremberg*. 91058 Erlangen, Germany.

Dechene, D.J., Jardali, A.E., Luccini, M., & Sauer, A. (2010). A Survey of Clustering Algorithms for Wireless Sensor Networks. *Department of Electrical and Computer Engineering, University of Western Ontario*, Ontario.

Halgamuge, M.N., Chan, T.K., & Mendis, P. (2009). Experiences of Deploying an Indoor Building Sensor Network. *Third International Conference on Sensor Technologies and Applications, in IEEE Computer Society.*

Haque, I.T. (2011). Location-Based Routing and Indoor Location Estimation In Mobile Ad hoc Networks. *Department of Computing Science, University of Alberta*, Edmonton Alberta.

Hill, J.L. (2003). System Architecture for Wireless Sensor Networks. *University of California, Berkeley*.

Idserda, J. (2004). TCP/IP Modelling in OMNeT++. *B-Assignment Telematics.* University of Twente, Netherlands.

Jangra, A., Richa, Sweti, & Priyanka. (2010). Wireless Sensor Network (WSN): Architectural Design Issues and Challenges. International Journal on Computer Science and Engineering (IJCSE). Vol. 02, No. 09. 3089-3094.

Kenyeres, J., Sajban, S., Farkas, P., & Rakus, M. (2010). Indoor Experiment with Wireless Sensor Network Apllication. *Department of Telecommunications, Slovak University of Technology, Slovakia, (MIRPO 2010).*

Krishnamachari, B. (2005). Networking Wireless Sensors. Cambridge University Press, New York.

Lewis, F.L. (2004). Wireless Sensor Networks. *Advanced Controls, Sensors, and MEMS Group, Automation and Robotics Research Institute, University of Texas at Arlington,* Jack Newell Blvd. S, Texas.

Loong, W.M. (2011). Wireless Mesh Network Simulation Using OMNeT++. SIM University School of Science and Technology.

Lu, J., Lu, D., & Huang, X. (2010). Channel Model for Wireless Sensor Networks in Forest Scenario. *Shenzhen Institutes of Advanced Technology Chinese Academy of Sciences,* Shenzhen, China.

Mallanda, C. D. (2005). Sensor Simulator: Simulation Framework for Sensor Networks. *Department of Computer Science and Engineering.* Kuvempu University.

Mallanda, C., Suri, A., Kunchakarra, V., Iyengar, S.S., Kannan, R., & Durresi, A. (2005). Simulating Wireless Sensor Network with OMNeT++. *Sensor Network Research Group, Department of Computer Science,* Louisiana State University, Baton Rouge, LA 2005, Version 1.

Panditharathne, C., & Sen, S.J. (2009). Energy Efficient Communication Protocols for Wireless Sensor Networks. *Department of Electronics and Communication Engineering, National Institute of Technology,* Rourkela, Orissa.

Park, S., Savvides, A., & Srivastava, M.B. (2000). SenSim: A Simulation Framework for Sensor Networks. Electrical Engineering Department, University of California, Los Angeles.

Sahni, S., & Xu, X. (2004). Algorithms for Wireless Sensor Networks. *Department of Computer and Information Science and Engineering.* University of Florida Gainesville, Florida, USA.

Salazar, A. (2010). Wireless Sensor Network Simulator. Office of Graduate Studies, Texas A&M University Corpus Cristi.

Selvakennedy, S., & Sinnappan, S. (2006). An Energy-Efficient Clustering Algorithm for Multihop Data Gathering in Wireless Sensor Networks. *Journal of Computers, Vol. 1, No. 1, Aprill 2006.* 40-47.

Schwartz, M. (2006). Mobile Wireless Communication. *2nd ed.* Cambridge University Press, New York.

Sohrabi, K., Minollil, D., & Znati, T.F. (2007). Wireless Sensor Network: Technology, Protocol and Application. Wiley Interscience.

Varga, A. (2005). OMNeT++ Discrete Event Simulation System. *Version 3.2. User Manual. Last updated 29 March, 2005.*

Velmani, P., & Ramar, K. (2011). Design and Implementation of Logical Topology in Sensor Network for an Industrial Stack Monitoring. *International Journal of Computer Applications (0975-8887), Volume 25 No. 7, July 2011.*

Yang, H., & Yang, S.H. (2009). Connectionless Indoor Inventory Tracking in Zigbee RFID Sensor Network. *Computer Science Department, Ioughborough University UK.*

Zhang, J., Chen, J., Xu, W., & Sun, Y. (2008). OMNeT++ Based Simulation for Topology Control in Wireless Sensor Network: A Case Study. *Institute of Industrial Process Control, Zhejiang University & College of Informatics and Electronics, Zhejiang Sci-Tech University,* Hangzhou, China. IEEE.

Zhang, Y., Fang, Z., Li, R., & Hu, W. (2009). The Design and Implementation of a RSSI-Based Localization System. *School of Computer Science and Technology, Jilin University,* Changchun, 130012, R.R China.

Zhao, J., & Govindan, R. (2003). Understanding Packet Delivery Performance in Dense Wireless Sensor Networks. Proc. 2nd, Embedded Networked Sensor Systems Conf. SenSys '03. (pp. 1- 13) ACM Press, New Yprk, USA.