DENIS LIMA DO ROSÁRIO

Uma Plataforma Experimental para Avaliação de Desempenho de Estimadores de Qualidade de Enlace em Rede de Sensores Sem Fio

FLORIANÓPOLIS
2010

UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE AUTOMAÇÃO E SISTEMAS

Uma Plataforma Experimental para Avaliação de Desempenho de Estimadores de Qualidade de Enlace em Rede de Sensores Sem Fio

Dissertação submetida à
Universidade Federal de Santa Catarina
como parte dos requisitos para a
obtenção do grau de Mestre em Engenharia
de Automação e Sistemas.

DENIS LIMA DO ROSÁRIO

Florianópolis, abril de 2010.

Uma Plataforma Experimental para Avaliação de Desempenho de Estimadores de Qualidade de Enlace em Rede de Sensores Sem Fio

Denis Lima do Rosário

‘Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Engenharia de Automação e Sistemas, Área de Concentração em *Controle, Automação e Sistemas*, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia de Automação e Sistemas da Universidade Federal de Santa Catarina.’

Aos meus pais.

AGRADECIMENTOS

Em primeiro lugar gostaria de agradecer a Deus, pois ele é o responsável por tudo que acontece na minha vida.

Em segundo lugar a minha família que apesar de distantes sempre me apoiaram muito, em especial aos meus pais, Jose Everaldo e Waldeci que, em todos os momentos, estiveram presentes nesta jornada de estudos. Também, as minhas irmãs, Danielle e Denise pela paciência e apoio que sempre me deram.

A Tainá que é uma pessoa muito especial para mim, mesmo longe de mim sempre me deu muito apoio e incentivo para continuar essa jornada de estudos.

Gostaria de agradecer ao meu orientador de mestrado Leandro Buss Becker pela orientação e por me ajudar a melhorar este trabalho e principalmente a redigir este documento.

Agradeço também aos orientadores no período em que estive no ISEP Mário Alves e Anis Koubâa, que colaboraram consideravelmente para a construção deste trabalho.

Gostaria de agradecer ao CNPq pelos recursos concedidos ao projeto 486250/2007. Agradeço também ao ISEP pelo apoio técnico e financeiro que tornou possível a construção desta ferramenta, em especial ao professor Eduardo Tovar.

Agradeço também a imprescindível colaboração das colegas associadas ao ISEP Nouha Baccour e Maissa Ben Jamâa, que não mediram esforços em nossas colaborações à distância. Também a todos os membros do ISEP que me ajudaram neste trabalho, em especial ao Ricardo Severino, Osvaldo Sousa e outros.

Agradeço também as pessoas que me ajudaram de alguma forma direta ou indiretamente na elaboração deste trabalho e que eu tenha esquecido.

Por ultimo, mas não menos importante aos amigos do DAS pelos momentos de descontração e pelo compartilhamento de conhecimento e aos meus amigos de Belém em especial ao Cloudo e Janny pelo incentivo e apoio durante estes dois anos de estádia em Santa Catarina.

Resumo extendido da Dissertação apresentada à UFSC como parte dos requisitos necessários para obtenção do grau de Mestre em Engenharia de Automação e Sistemas.

Uma Plataforma Experimental para Avaliação de Desempenho de Estimadores de Qualidade de Enlace em Rede de Sensores Sem Fio

Denis Lima do Rosário

Abril/2010

Orientador: Leandro Buss Becker, Dr.
Co-orientador: Mário Ferreira Alves, Dr.
Área de Concentração: Controle, Automação e Sistemas
Linha de Pesquisa: Sistemas Computacionais
Palavras-chave: Redes de Sensores Sem Fio, Plataformas Experimentais e Estimadores de Qualidade de Enlace
Número de Páginas: xxviii + 86

Uma rede de Sensores Sem Fio (RSSF) consiste em dispositivos de baixo custo capazes de executar aplicações individuais e de se comunicarem entre si utilizando enlace sem fio. Estas tecnologias de rede dependem de diversos protocolos distintos, como Controle de Acesso ao Meio (MAC), roteamento e controle de topologia. Neste contexto, estimadores de qualidade de enlace (do inglês *Link Quality Estimator* - LQE) mostram-se como componentes essenciais para serem utilizados por estes protocolos.

A realização de experimentos reais em RSSF permite aos pesquisadores obterem resultados mais precisos se comparado com estudos de simulação. Contudo, realizar experimentos em um ambiente real exige o uso de uma plataforma experimental apropriadas, chamada de *testbed*. Neste sentido, diversos *testbeds* foram propostos para suportar experimentos com RSSF porem existem algumas características desejadas para realizar experimentos para avaliação de estimadores de qualidade de enlace que não são contemplada em tais *testbeds*. Como exemplo podemos citar a coleta de todos os pacotes recebidos e enviados pelos nodos, o suporte à escolha de diversos parâmetros de redes para os experimentos, o suporte à reprogramação de nodos, a disponibilização de análises *off-line* e informação sobre a localização dos sensores.

Por estes motivos nesta dissertação de mestrado propõem-se o desenvolvimento de um novo *testbed*, capaz de suportar as características mencionadas acima. O *testbed* proposto consiste em uma plataforma de *hardware* e uma ferramenta de *software*.

A plataforma de *hardware* é formada por componentes de *hardware* disponíveis no mercado, que são um conjunto de nodos sensores conectados a uma estação de controle por meio de uma combinação de cabos USB e Hubs USB ativos, constituindo assim um *backbone* USB, este *backbone* USB é utilizado para que os nodos possam reportar a estação de controle todos os pacotes enviados e recebidos. Já a estação de controle pode reprogramar os nodos, enviar e receber mensagens de inicio e fim de transmissão.

A ferramenta de *software* consiste em uma Interface Gráfica de Usuário, a qual foi desenvolvida em Java e prove algumas funcionalidades para o usuário tais como: detecção automática dos nodos conectados a estação de controle, reprogramação e controle dos nodos, configuração de alguns parâmetros de rede e armazenamento dos dados coletados em um banco de dados, dessa forma permitindo à realização de análises *off-line*.

Assim que a aplicação é iniciada, automaticamente é detecto os nodos conectados à estação de controle e mostra uma lista de nodos conectados para o usuário. O usuário seleciona os nodos e então pode reprogramá-los. O usuário também pode selecionar alguns parâmetros de rede para ser usado no experimento, tais como: (i) padrão de tráfego: rajada ou sincronizado; (ii) canal de rádio; (iii) potência de transmissão; (iv) tamanho do pacote; (v) número máximo de retransmissões; (vi) intervalo entre mensagens; (vii) número de pacotes a serem enviados. Durante o experimento, os dados coletados pelos nodos são reportados à estação de controle e então armazenados em um banco de dados.

Para ilustrar a utilidade do *testbed* proposto apresentamos um estudo de caso, onde se faz um estudo comparativo de diferentes LQEs para RSSF. Para alcançar tal objetivo, foram conduzidos diversos experimentos, fazendo variações nas configurações da rede de forma a interferir nos LQEs. Não foi considerado qualquer fator externo, como roteamento e colisão. O objetivo dos experimentos foi comparar a propriedade de estabilidade e confiabilidade dos LQEs para diferentes configurações de rede. Tal informação é útil para ajudar no projeto de camadas mais altas de protocolos, tornando-os ciente de qual(is) LQEs é(são) mais resistente(s) à flutuação na qualidade do enlace.

Por fim, é importante ressaltar as contribuições do *testbed* proposto neste estudo comparativo de LQEs. O *testbed* ajuda o usuário a selecionar e reprogramar os nodos, tarefa esta que foi realizada dezenas de vezes ao longo do experimento. Também foi preciso executar um conjunto de experimentos mudando os parâmetros de rede, então o *testbed* foi útil para ajudar a selecionar esses parâmetros. Além disso, é possível coletar e armazenar todos os pacotes enviados e recebidos; todos os parâmetros de rede selecionados e todas as informações dos experimentos são armazenados em um banco de dados. Estas informações são úteis para realizar a análise *off-line* dos dados; a informação de coordenada dos nodos nos ajuda a desenhar a topologia de rede e analisar a qualidade dos enlaces baseada na distância entre os nodos.

A ferramenta proposta foi desenvolvida em parceria com o grupo de pesquisa CISTER do Instituto Superior de Engenharia do Porto, com o objetivo de coletar dados para realizar um estudo comparativo com os estimadores de qualidade de enlace. O *testbed* proposto é *open-source* e esta disponível em [38].

Este volume está dividido em seis capítulos, conforme descrição a seguir.

O capítulo 1 (*Introduction*) apresenta o tema de Redes de Sensores e a motivação para a criação de um novo *testbed* para avaliação de estimadores de qualidade de enlace. São apresentados também os objetivos e as contribuições do trabalho, juntamente com o contexto de colaboração internacional em que o mesmo se encontra inserido.

Já o capítulo 2 (*WSN Technologies*) apresenta as tecnologias relacionadas a rede sensores tais como os dispositivos de RSSF mais utilizados, os padrões para RSSF, os sistemas operacionais e os estimadores de qualidade de enlace.

As características dos principais *testbeds* existentes para RSSF, os quais baseiam-se em um conjunto de nodos conectados a um computador através de cabos USB ou serial, são apresentados no capítulo 3 (*Related Works*). Também são apresentados as características desejadas em um *testbed* para realizar experimentos para avaliar estimadores de qualidade de enlace, características essas que foram incluídas no *testbed* proposto nesta dissertação de mestrado.

O *testbed* proposto, chamado de RadiaLE Testbed é apresentado no capítulo 4 (*The RadiaLE Testbed*). Neste capítulo são mostrados os detalhes da arquitetura de *hardware*, da ferramenta de *software* proposta para realizar experimentos, e as limitações do *testbed* proposto.

Os experimentos realizados e os resultados obtidos para ilustrar a utilidade e usabilidade do RadiaLE Testbed são apresentados no capítulo 5 (*Case Study: LQEs Performance Evaluation*).

Por fim, o capítulo 6 (*Conclusion*) apresenta as conclusões e trabalhos futuros do trabalho.

Abstract of Dissertation presented to UFSC as a partial fulfillment of the requirements for the degree of Master in Automation and Systems Engineering.

Benchmarking Testbed for the Performance Evaluation of Link-Quality Estimators in Wireless Sensor Network

Denis Lima do Rosário

April/2010

Advisor: Leandro Buss Becker, Dr.
Co-advisor: Mário Ferreira Alves, Dr.
Area of Concentration: Control, Automation and Systems
Research Area: Computation Systems
Key words: Wireless Sensor Networks, Testbeds, and Link Quality Estimators
Number of Pages: xxviii + 86

Performing real experimentation in wireless sensor networks (WSN) instead of working with simulations allows researchers to obtain much more accurate results. This can be applied, for instance, to understand and evaluate new MAC protocols, routing algorithms, and also link quality estimators (LQEs). Real experimentation is, however, only feasible to be applied if using proper computational tools, normally called testbeds. For this reason, we propose a new testbed to perform experiments for evaluation of Link Quality Estimators in WSN. Our testbed includes (i) use of commercial-of-the-shelf hardware components for performing experiments and collecting related data and (ii) a software tool to control and analyze the experiments. We developed a case study that uses the proposed Testbed to make the performance evaluation of several LQEs. Despite the fact that the study helped us to make small adjustments in the tool, the obtained results were of paramount importance for the creation of a new LQE in the context of a related work.

Contents

List of Acronyms

ACERVO	communication ArChitecturE for Real-time mobile cooperatiVe applicatiOns
APL	Application layer
APS	Application Support
ASL	asymmetry
ASNR	channel quality
CAP	Contention Access Period
CFP	Contention-Free Period
CC	Correlation Coefficient
CCA	Clear Channel Assessment
CDF	Cumulative Distribution Function
CSMA-CA	Carrier Sense Multiple Access with Collision Avoidance
CV	Coefficient of Variation
DataAnlApp	Data Analysis Application
ED	Energy Detection
ExpCtrApp	Experiment Control Application
ETX	Expected Transmission Count
F-LQE	Fuzzy Link Quality Estimator
GUI	Graphical User Interface
GTS	Guaranteed Time Slot
Id	Identifier
IEEE	Institute of Electrical and Electronics Engineers

IPI	Inter-Packet Interval
LQE	Link Quality Estimator
LQI	Link Quality Indicator
LR-WPAN	Low-Rate Wireless Personal Area Networks
MAC	Medium Access Control
NWK	Network layer
OS	Operating System
OSI	Open System Interconnection
PRR	Packet Received Rate
PHY	Physical Layer
RNP	Required Number of Packet retransmissions
RSSF	Redes de Sensores Sem Fio
RSSI	Received Signal Strength Indicator
Rtx	retransmissions
RTOS	Real-Time Operating System
SCALE	A tool for Simple Connectivity Assessment in Lossy Environments
SNR	Signal-to-Noise Ratio
SPRR	packet delivery
SF	stability
SWAT	The Stanford Wireless Analysis Tool
TinyOS	Tiny Microthreading Operating System
TWIST	TKN Wireless Indoor Sensor network Testbed
UML	Unified Modeling Language
USB	Universal Serial Bus
WMEWMA	Window Mean Exponentially Weighted Moving Average
WPAN	Wireless Personal Area Network
WSN	Wireless Sensor Network
ZDO	ZigBee Device Object

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Wireless sensor networks (WSNs) consist of many sensing devices which can run individual applications and communicate with each other using a wireless link [24]. To support such applications, several distinct network protocols were proposed in the literature, such as Medium Access Control (MAC) [30], routing [31], and topology control [27]. Link quality estimators (LQEs) [16] is a fundamental building block to design higher layer protocols, such as topology control, routing, and mobility management protocols. For instance, usually routing protocols rely on LQEs as a support mechanism to select the most stable routes for data delivery.

The first design step of new applications and protocols is normally its simulation. There exists reasonable tools for simulation of WSN protocols, such as TOSSIM [26], OMNET [39], and OPENET [33]. The drawback of simulation work is that it can be not very realistic, such as in real experimentation. Consequently, in simulation researches are frequently forced to make artificial assumptions about traffic, topologies, environment, and failure patterns. The next step is the implementation and evaluation of applications and protocols in real environments. Therefore it is required real experimentation, which implies the use of real hardware, deployed in existing environments, and using realistic experimental setups.

On the other hand, the validation of WSNs protocols in real environments can be frustrating. This is due the fact that it is necessary to reprogram several times a large number of nodes, which are normally deployed in a very large area. Experimental work should include both data logger and data analysis, which are very difficult to perform without a tool that supports these features.

In this context using a testbed is very helpful. A testbed is a platform for experimentation of large projects. Testbeds allow for rigorous, transparent, and replicable testing of scientific theories, computational tools, and new technologies. In the literature, several testbeds were proposed to use with WSN, such as SCALE [9], MoteLab [41], TWIST [19], SWAT [32], Mirage [12], and others.

These testbeds have interesting features but they present several drawbacks when considering the evaluation of LQEs. These testbeds lack many expected features, such as logging all packet from sender and receiver side, adjusting the parameters for each experiment, efficient support for sensor nodes (motes) reprogramming, off-line analyzes support, awareness about motes locations, and allowing direct interaction with motes.

To overcome the mentioned deficiencies in existing testbeds, in this master thesis it is proposed the design and implementation of a new testbed to perform experiments for evaluation of LQES using WSNs, called RadiaLE testbed. The RadiaLE testbed is part of the RadiaLE framework [38], which has a broader scope since it also integrates a tool to perform the off-line analysis of data.

The proposed testbed consists in a software tool and in a hardware platform. The software tool includes a Graphical User Interface (GUI) that allows users to make: motes selection, programming, and control, network configuration, collect useful information from each data packet, and perform off-line analyzes. The hardware platform consists in the use of commercial-of-the-shelf components, for performing experiments and collecting data from the motes without interfering in the wireless communications. The hardware components include a set of wireless sensor nodes connected to a laptop PC via combination of a USB cables and active USB hubs.

1.2 Research Context

This master thesis was developed in collaboration between the Department of Automation and Systems (DAS) of the Federal University of Santa Catarina (UFSC), CISTER Research Unit at ISEP/IPP (*Instituto Superior de Engenharia do Porto/ Instituto Politécnico do Porto*) and the ReDCAD research unit at ENIS (*Ecole Nationale d'Ingénieurs de Sfax, University of Sfax*). Other institutions have partially contributed to the development of this master thesis, which are: Al-Imam Mohamed bin Saud University and PRINCE Research Unit (*Institut Spérieur d'Informatique et des Technologies de Communication de Hammam Sousse*), University of Sousse.

This collaboration was in the context of the Ph.D. work of Nouha Baccour, which is performed in collaboration between ReDCAD research unit at ENIS and CISTER Research Group at ISEP/IPP. The objective of this thesis is to provide efficient link quality estimators for wireless sensor networks. One practical objective in Nouha Baccour's PhD is the experimentation of LQEs using real deployment to validate the performance of a newly proposed fuzzy link quality estimator (F-LQE) by comparing its performance to those of existing LQEs. For that purpose, Nouha Baccour has designed RadiaLE, a benchmarking testbed that enables the performance evaluation of LQEs in Wireless Sensor Networks. RadiaLE comprises the hardware components of the WSN testbed (TelosB nodes, USB cables/hubs) and a software tool for setting up and controlling the experiments and also for analyzing the collected data, allowing for LQEs evaluation.

This master thesis is also related to the CNPq (*Conselho Nacional de Desenvolvimento Científico e Tecnológico*) project named ACERVO: communication ArChitecturE for Real-time mobile cooperatiVe applicatiOns [29] [17]. This project is directly related with the Ph.D. work of Marcelo Maia Sobral, which is performed at PGEEL/UFSC. Its goal is to create a real-time communication architecture for mobile and autonomous systems.

The implementation of RadiaLE functionalities as well as its deployment at Porto has been done jointly by Maissa Ben Jamâa and Denis Lima do Rosário in the context of their Masters works. The work of Maissa was mainly the implementation of a MATLAB application for analyzing the experimental data and generating graphs and statistics. On the other hand, the work of Denis was mainly the deployment of RadiaLE at Porto, namely (i) the deployment and the establishment of the circular network topology (technology specification (hubs, and cables), and specifies the hubs and cables layout), and testing/troubleshooting the RadiaLE code, and (ii) performing several indoor and outdoor experiments at Porto . A joint work between Maissa and Denis was the implementation of a Java application for the experiment control.

The current master thesis is written by Denis Lima do Rosário as a description of this joint work for both personal contributions and those being achieved in collaboration.

1.3 Goals of the Master Thesis

The main goal of this master thesis is to make the design and implementation of a new testbed to perform experiments for evaluation of LQEs for WSN. This testbed should allow to: (i) select, program, and control the motes, (ii) configure a set of parameters for the experiment, (iii) gather and store useful information from each packet during the experiment, and (iv) perform off-line analysis of collected data. As specific goal, we aim to use this testbed to perform experiments that will serve to collect data, and make an accurate analysis of existing LQEs for wireless sensor networks.

1.4 Structure of this Master Thesis

The remainder parts of this master thesis are structured as follows. Chapter 2 introduce some related technologies to WSN.

Chapter 3 details the most relevant testbeds for WSN and also includes a comparison between these testbeds and the proposed one.

Chapter 4 presents the RadiaLE testbed, including a description of design and implementation of the hardware platform and software tool.

The experiments performed using the proposed testbed and the analyze of the obtained results using the RadiaLE testbed are presented in Chapter 5.

The master thesis concludes with Chapter 6, which summarizes the main contributions and results of this master thesis, and identifies topics for future works.

Chapter 2

WSN Technologies

In this chapter we present some technologies related to WSNs, including devices, Wireless network standards, operation systems, and LQEs.

2.1 Introduction

A WSN consists of spatially distributed autonomous sensors nodes used to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants. This kind of network may be deployed in almost any type of environment, due to the fact of its small size, easy communication, and low cost.

The development of WSNs was originally motivated by military applications such as battlefield surveillance. Now they are 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.

Typically, a sensor node is equipped with a radio transceiver or other wireless communications device, a small microcontroller, an energy source (i.e. a battery), and some sensor. 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 hundreds of dollars to a few pennies, depending on the size and the complexity of individual sensor nodes. Size and cost constraints on sensor nodes result in corresponding constraints on resources such as energy, memory, computational speed, and bandwidth.

A sensor network normally constitutes a wireless ad-hoc network, meaning that each sensor supports a multi-hop routing algorithm, for this reason a sensor network can have a large communication range.

WSNs have a rich set of application, normally involving some kind of monitoring, tracking, or controlling. Have some specific applications such as habitat monitoring, object tracking, nuclear reactor control, fire detection, and traffic monitoring. In a typical application, the WSN is deployed in an environment to the sensor nodes collect data. Follow some examples of applications for WNs.

WSN can be applied to medical application for monitoring patients such as presented in [28], to monitoring hard environment such as for volcanic eruptions [40], and to monitor structural integrity of a buildings/tower as show [8].

2.2 Typical WSN platforms

In this subsection we present the mains features and the technical characteristics of some well know WSN devices.

1. **TelosB [37]** is a sensor node developed by Crossbow (figure 2.1). This mote uses the IEEE 802.15.4 compliant RF transceiver developed by Chipcom (CC2420) [11]. The mote already has integrated an USB interface used for programming and data collection. For outdoor environment this mote has a communication range of 75m to 100m and for indoor 20m to 30m. Follows some technical details about this sensor node:

   - TI MSP430 16-bit microcontroller [21];
   - CC2420 RF transceiver [11], onboard antenna;
   - 48 KB of program flash memory;

- 16 KB of EEPROM;
- 250 kbps data rate;
- Integrated light, temperature and humidity sensor.

Figure 2.1: TelosB mote [37]

Figure 2.2: MICAz mote [36]

The Tmote Sky is a wireless sensor board that was sold by Sentilla and developed by Moteiv Corporation. It is the last version of TelosB that has the same design but is sold by Crossbow.

2. **MICAz[36]** this sensor node is also developed by Crossbow (figure 2.2). As MICAz uses the same radio transceiver of TelosB, it has the same communication rage. Follows some technical details about MICAz:

   - ATMEL ATmega128L 8-bit microcontroller [3];
   - CC2420 RF transceiver [11];
   - 128 KB of program flash memory;
   - 4 KB of EEPROM;
   - 250 kbps data rate;
   - Expansion connector for light, temperature, barometric pressure, and other Crossbow sensor boards.

   To programm a MICAz mote it is necessary to use an interface board, such as MIB510 (figure 2.3) [34] or MIB520 (figure 2.4) [35]. These boards provide a serial/USB interface for both programming and data communications (i.e. used as base stations). These boards are very similar, except for the fact that MIB510 has a serial RS-232 interface and the MIB520 has an USB interface.

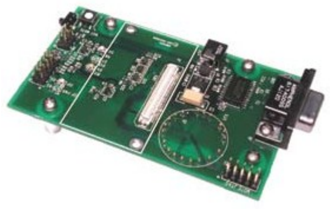


Figure 2.3: MIB510 Serial Interface Board [34]

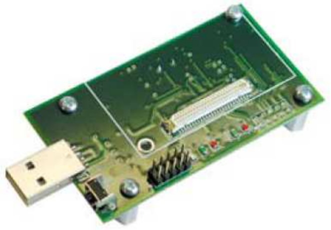


Figure 2.4: MIB520 USB Interface Board [35]

2.3 Wireless Network Standards

Our interest here relies in the standard IEEE 802, which refers to a family of standards dealing with local area networks and metropolitan area networks. The services and protocols specified in IEEE 802 refer to the two lowers layers (Data Link and Physical (PHY)) from the seven-layer OSI (Open System Interconnection) networking reference model. The related standards are the following.

- IEEE 802.3 - Ethernet
- IEEE 802.11 - Wireless LAN (Wi-Fi)
- IEEE 802.15 - Wireless Personal Area Network (WPAN)
  - IEEE 802.15.1 - Bluetooth certification
  - IEEE 802.15.4 - Low-rate WPAN certification
- IEEE 802.16 - Broadband Wireless Access (Wimax)

2.3.1 IEEE 802.15.4

IEEE 802.15.4 specifies the low-data-rate (up to 250 kbps) wireless connectivity with fixed, portable, and moving devices with very limited energy consumption requirements, operating in a shorter distance (typically 10m, but can reach 100m) and accept star or peer-to-peer topology [1].

The standard architecture is show in figure 2.5. It specifies the PHY and MAC sublayer for low-rate wireless personal area networks (LR-WPANs).

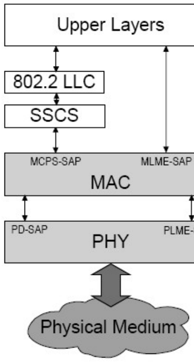


Figure 2.5: IEEE 802.15.4 Standard Architecture [1]

2.3.2 Physical Layer

IEEE 802.15.4 standard specifies that the PHY is responsible for services such as activation and deactivation of the radio transceiver, Energy Detection (ED) within the current channel, Link Quality Indicator (LQI) for a received packets, channel frequency selection, Clear Channel Assessment (CCA) for Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA), and data transmission and reception across the physical medium according the specific modulation and spreading technique. Follows more details about each service provided by PHY.

1. **Activation and Deactivation of the Radio Transceiver:** the radio transceiver has three states: transmitting, receiving or sleeping. Upon

request of the MAC sub-layer, the radio is turned to ON or OFF.

2. **Energy Detection:** is an estimate of the received signal power within the bandwidth of IEEE 802.15.4 channel. No attempt is made to identify or decode signals on the channel. It is used by a network layer as part of a channel selection algorithm.

3. **Link Quality Indicator:** is a measurement characterization of a strength and/or quality of a received packet. The minimal and maximum value of LQI are associated with the lowest and highest quality respectively, and LQI values in between should be uniformly distributed between these two limits.

4. **Clear Channel Assessment:** is performed according to at least one of the following three methods: (i)*Energy above threshold* it report a busy medium upon detecting any energy above the ED threshold. (ii)*Carrier sense only* it report a busy medium only upon the detection of a signal compliant with this standard with the same modulation and spreading characteristics of the PHY. This signal may be above or below the ED threshold. (iii)*Carrier sense with energy above threshold* it report a busy medium by detect a signal with the modulation and spreading characteristics of this standard and the energy above the ED threshold, where the logical operator may be AND or OR.

5. **Channel Frequency Selection:** the standard define a total of 27 channels (from 0 to 26), they are available across the three frequency bands (16 channels to 2450 MHz band, 10 to 915 MHz band, and 1 to 868 MHz band). PHY should be able to change the channel when requested by a higher layer.

2.3.3 MAC Sublayer

The IEEE 802.15.4 standard defines that the WPAN can operate in the follow modes: beacon-enabled or non beacon-enabled. The network that requires synchronization or support for low-latency devices, such as PC peripherals, it is recommend use the beacon-enabled mode, otherwise is recommend use the non beacon-enabled mode.

For the beacon-enabled mode a coordinator defines the format of the superframe. The superframe is divided in 16 slots that are used for the nodes to send its packets. Beacons are sent by the coordinator to delimit the bound of a superframe and for synchronization. Contention Access Period (CAP) is a period between two beacons and during this period the devices competes with other devices using a slotted CSMA-CA mechanism. The superframe has an active and an inactive period, and the coordinator can enter in a low-power mode during the inactive period (figure 2.6 shows the structure of a superframe in a 802.15.4 standard). For the non beacon-enabled mode the nodes simply transmits its data frame, using unslotted CSMA-CA and without synchronization with the coordinator.

The GTS (Guaranteed Time Slots) mechanism allows a device to access the medium without contention in the CFP (Contention-Free Period) period, for applications that require guaranteed bandwidth. The GTSs always appears at the end of the active superframe starting at a slot boundary that immediately follows the CAP, as shown in Figure 2.7. The coordinator may allocate up to seven of these GTSs, and a GTS may use more than one slot period. However, a sufficient portion of the CAP remains for contention-based access of other networked devices or new devices wishing to join the network. Each device transmitting in a GTS ensures that its transaction is completed before the time of the next GTS or the end of the CFP [1].

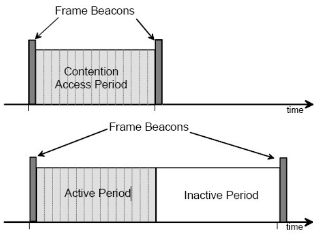


Figure 2.6: Superframe structure [1]

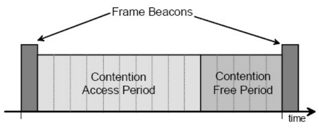


Figure 2.7: structure with GTSs [1]

2.3.4 ZigBee

The ZigBee protocol defines the network layer (NWK) and application layer (APL) from the OSI model and uses the PHY and MAC layer defined by the standard 802.15.4, as shown in figure 2.8.

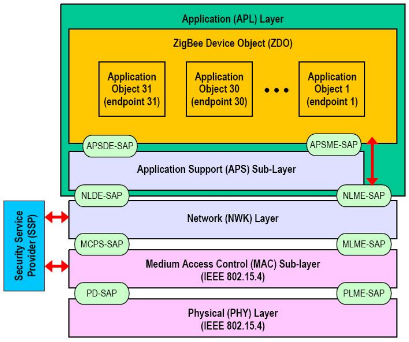

Application (APL) Layer
ZigBee Device Object (ZDO)
Application Object 31 (endpoint 31)
Application Object 30 (endpoint 30)
Application Object 1 (endpoint 1)
APSDE-SAP
APSME-SAP
Application Support (APS) Sub-Layer
NLDE-SAP
NLME-SAP
Network (NWK) Layer
MCPS-SAP
MLME-SAP
Medium Access Control (MAC) Sub-layer (IEEE 802.15.4)
PD-SAP
PLME-SAP
Physical (PHY) Layer (IEEE 802.15.4)
Security Service Provider (SSP)

Figure 2.8: ZigBee Architecture [2]

The NWK layer of ZigBee has the following responsibilities: (i) starting a network: ability to establish a new network; (ii) joining and leaving a network: ability to gain membership (join) or relinquish membership (leave) a network); (iii) configuring a new device: ability to sufficiently configure the stack for operation as required; (iv) addressing: ability to the ZigBee coordinator assign to the addresses for devices joining the network; synchronization within a network: ability to achieve synchronization with another device either through tracking beacons or by polling; (v) security: applying security to outgoing frames and removing security to terminating frames; (vi) routing: ability to routing frames to their intended destinations.

The ZigBee APL layer consists of the APS (Application Support) sublayer, the ZDO (ZigBee Device Object) and the manufacturer-defined application objects. The responsibilities of the APS sub-layer includes maintaining tables for binding, which is the ability to match two devices together based on their services and their needs, and forwarding messages between bound devices. Another responsibility of the APS sub-layer is discovery, which is the ability to determine which other devices are operating in the personal operating space of a device. The responsibilities of the ZDO include defining the role of the device within the network (e.g., ZigBee coordinator or end device), initiating and/or responding to binding requests and establishing a secure relationship between network devices. The manufacturer-defined application objects implement the actual applications according to the ZigBee-defined application descriptions [2].

2.4 Link Quality Estimators

LQEs is a fundamental building block for WSNs for the design of several different mechanisms and protocols in WSN. The accuracy of the LQE greatly impacts the efficiency of these protocols. Several LQEs have been reported in the literature, as further described.

1. **PRR (Packet Received Rate)**: is defined as the number of successfully received packets (Received) over the number of transmitted packets (Transmitted), for each window of *W* received packets. PRR can be calculated as follows:

$$PRR(W) = \frac{received}{transmitted} \tag{2.1}$$

2. **RNP (Required Number of Packet retransmissions) [10]:** counts the average number of packet retransmissions required before a successful reception. It is computed as the number of transmitted and retransmitted packets (Transmission) divided by the number of successfully received packets (Received), minus 1 to exclude the first packet transmission. This metric is evaluated at the sender side for each *W* retransmitted packets, as show the follow equiation.

$$RNP(W) = \frac{Transmission}{Received} - 1 \tag{2.2}$$

3. **WMEWMA (Window Mean Exponentially Weighted Moving Average) [42]:** applies filtering on PRR to smooth it, thus providing a metric

that resists to transient fluctuation of PRRs, yet is responsive to major link quality changes. WMEWMA can be calculated as follows:

$$WMEWMA(\alpha, W) = \alpha * WMEWMA + (1 - \alpha) * PRR \quad (2.3)$$

Where $\alpha \in [0 \ldots 1]$ controls the smoothness.

4. **ETX (Expected Transmission Count) [13]:** incorporates the effects of link loss ratios, asymmetry in the loss ratios between the two directions of each link, and interference among the successive links of a path. The ETX of a link is the predicted number of data transmissions required to send a packet over that link, including retransmissions. The ETX of a link is calculated using the PRR of the forward link (PRR$_{forward}$) and the PRR of the backward link (PRR$_{backward}$). It is computed as the inverse of the product of the PRR$_{forward}$ and the PRR$_{backward}$, as given by equation (2.4).

$$ETX(w) = \frac{1}{PRR_{forward} * PRR_{backward}} \quad (2.4)$$

5. **Four-bit [16]:** is a sender-initiated estimator, already implemented in TinyOS, that approximates the packet retransmissions count. Such as ETX, four-bit considers link asymmetry property. It combines two metrics (i) estETX$_{up}$, as the quality of the unidirectional link from sender to receiver, and (ii) estETX$_{down}$, as the quality of the unidirectional link from receiver to sender. The estETX$_{up}$ is exactly the RNP metric and estETX$_{down}$ approximates RNP as the inverse of WMEWMA, minus 1. The combination of the two metrics is performed through the EWMA filter as follows:

$$Four - bit(w_a, w_b, \alpha) = \alpha * four - bit + (1 - \alpha) * estETX \quad (2.5)$$

Where estETX corresponds to estETX$_{up}$ or estETX$_{down}$: given w_a the beacon-driven estimation window and w_p the data-driven estimation window; at w_a received packets, the sender derives the four-bit estimate by replacing estETX for estETX$_{down}$ in equation (2.5). At w$_p$ transmitted/re-transmitted data packets, the sender derives the four-bit estimate by replacing estETX for estETX$_{up}$.

6. **F-LQE (Fuzzy Link Quality Estimator) [7]:** combines four link quality properties namely, packet delivery (SPRR), asymmetry (ASL), stability (SF), and channel quality (ASNR). SPRR is related to the capacity of the link to successfully deliver data, ASL is the difference in connectivity between the uplink and the downlink, SF is the variability level of the link, and the ASNR can be evaluated through the measure of SNR. Each of these proprieties is defined as a natural language of fuzzy logic. The overall quality of the link is specified as a fuzzy rule whose evaluation returns the membership of the link in the fuzzy subset of good links. Values of the membership function are smoothed using EWMA filter to improve stability. The goodness (i.e. high quality) of a link is characterized by the following rule:

   IF the link has *high packet delivery* AND *low asymmetry* AND *high stability* AND *high channel quality* THEN it has *high quality*.

   Here, high packet delivery, low asymmetry, high stability, high channel quality, and high goodness are linguistic values for the fuzzy variables. Using and-like compensatory operator, the above rule translates to the following equation of the fuzzy measure of the link *i* high quality.

$$\begin{aligned}\mu(i) = \beta.min(\mu SPRR(i), \mu ASL(i), \mu SF(i), \mu ASNR(i)) +\\ (1-\beta).mean(\mu SPRR(i), \mu ASL(i), \mu SF(i), \mu ASNR(i))\end{aligned} \quad (2.6)$$

   Where μ (*i*) is the membership in the fuzzy subset of high quality links. The parameter β is a constant in [0..1]. μSPRR, μASL, μSF, and μASNR represent membership functions in the fuzzy subsets of high packet delivery, low asymmetry, low stability, and high channel quality, respectively.

2.5 Operating Systems

An Operating System (OS) is the software that provides an interface between the hardware and the applications. The OS is in charge to react to events and also to handle access to memory, CPU, and hardware peripherals. In this section we present some typical OS used in WSN.

1. **TinyOS (Tiny Microthreading Operating System) [20]** is the most popular OS designed for WSNs. TinyOS started as a collaboration between the University of California, Berkeley in co-operation with Intel Research and Crossbow Technology, and has since grown to be an international consortium, the TinyOS Alliance. It is an open-source embedded OS written in the nesC programming language [18], which is a language for programming structured component-based applications. nesC is an extension to the C programming language and is designed to express the structuring concepts of TinyOS.

   TinyOS has an event-driven execution model and the component library includes network protocols, distributed services, sensor drivers, and data acquisition tools. TinyOS has support to a large range of platforms and numerous sensor boards. TinyOS applications are built out of components and interfaces. Components are connected to each other using interfaces. TinyOS provides interfaces and components for common abstractions such as packet communication, routing, sensing, actuation and storage.

2. **Nano-RK [15]** is a real-time operating system (RTOS) from Carnegie Mello University designed for WSNs. Nano-RK is fully preemptive and supports fixed-priority preemptive multitasking for ensuring that task deadlines are met, along with support for CPU, network, as well as, sensor and actuator reservations. Tasks can specify their resource demands and the operating system provides timely, guaranteed and con-

trolled access to CPU cycles and network packets. Together these resources form virtual energy reservations that allow the OS to enforce system and task level energy budgets. Nano-RK is open source, written in C and currently runs on the FireFly sensor networking platform, and MicaZ motes.

2.6 Concluding remarks

WSN consists of distributed autonomous sensors nodes that communicate using a wireless link. The sensor nodes normally use an OS to provide an interface between the hardware and the applications, facilitating nodes programming. The sensor nodes architecture follows the IEEE 802.15.4 standard that defines the Data Link and PHY from the seven-layer OSI networking reference model. Some application follows the ZigBee protocol that defines the NWK and APL from OSI model.

The communication protocol between sensor nodes relies on the use of LQE to proper estimate the link quality. The protocol efficiently depends on correct estimates, independent of the region of connectivity of the nodes (connected, transitional or disconnected).

Chapter 3

Related Works

We are interested in testbeds that all sensor nodes are connected to the PC using USB or serial cables. Has some application that relies on the use of a base station such as [40] and [8].

In this chapter it presented the existing testbeds for WSN. First it is described the main features from each of analyzed testbed. Afterwards we show the advantages of our proposed testbed in comparison with such testbeds.

3.1 Mirage: A Microeconomic Resource Allocation System for Sensornet Testbeds

The main goal of Mirage [12] is to allocate testbed resources by users competing in a maximally efficient manner in terms of aggregated utility delivered to users. The resources are allocated using a repeated combinatorial auction within a closed virtual currency environment. Users compete for testbed resources by submitting bids that specify resource combinations of interest in space/time along with a maximum value amount that the user is willing to pay. A combinatorial auction runs periodically to determine an efficient set of winning bids and subsequently provides users with controlled physical access to the relevant nodes.

The implementation is composed by three types of components: clients, server, and front-end machine that provide controlled physical access to the testbed, as shown in figure 3.1. Clients provide users with secure, authenticated command-line (the *mirage* program) and web-based access to a server (*mirage*) which implements a logical combinatorial auction, bank, and resource discovery service. The server accepts secure, authenticated XML-RPC requests using the SSL protocol with persistent state stored in a PostgreSQL database.

The users submit bids to the auction process, which consists of two-phases (see figure 3.2). First, using the resource discovery service, the user finds the candidate nodes that meet their constraints. Afterwards, based on the nodes identified during the first step, the user submits a bid in the auction process. The equation (3.1) specifies the bid b_i.

$$b_i = (v_i, s_i, t_i, d_i, f_{min}, f_{max}, n_i, ok_i) \tag{3.1}$$

Where b_i indicates a combination of n_i motes from the set ok_i, obtained through the resource discovery, for a duration of d_i hours, starting at a time between s_i and t_i, and with a frequency in the range of $[f_{min}, f_{mx}]$. The user is also willing to pay up to v_i units of virtual currency for these resources.

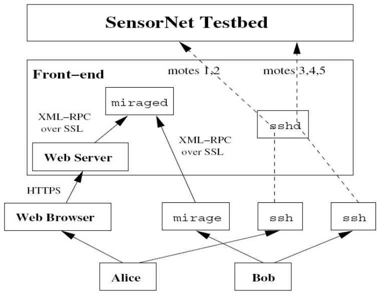


Figure 3.1: Mirage Implementation [12]

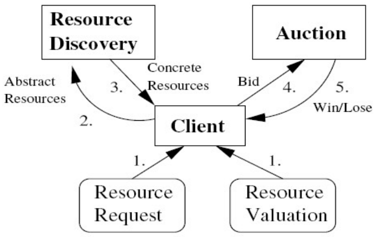


Figure 3.2: Bidding and Acquiring Resources [12]

Follow an example of a bid. Suppose that using the resource discovery the user finds 128 MICA2 motes meeting the desired resource specification and valued the allocation at 99 units of virtual currency. For this situation an example of bid can be: bi = (99, 0, 20, 4, 423, 443, 64, list of 128 motes), which means ”any 64 MICA2 motes, which have both a temperature and a humidity sensor, operating in a frequency in the range of [423 MHz, 443 MHz], for 4 consecutive hours, anytime in the next 24 hours”.

3.2 MoteLab: Wireless Sensor Network Testbed

MoteLab [41] is a web-based sensor network testbed, it consists of a set of permanently deployed sensor network nodes connected to a central server which support reprogramming and data logging while providing a web interface for creating and scheduling jobs on the testbed. MoteLab accelerates debugging and development by automating data logging, allowing the performance of sensor network software to be evaluated off-line. Additionally, by providing a web interface MoteLab allows to user both local and remote access to the testbed. Figure 3.3 depicts the testbed architecture. Follows an explanation about each component of the architecture.

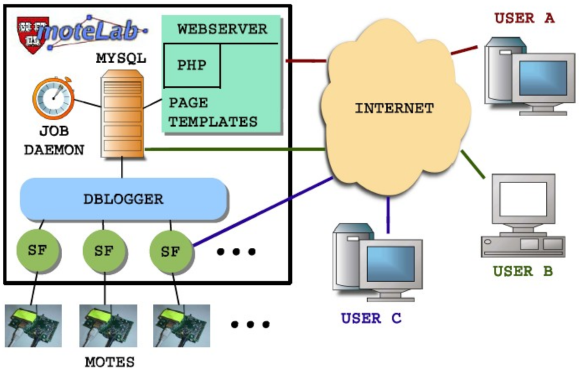


Figure 3.3: MoteLab Architecture [41]

- **MoteLab Hardware:** the management software handles a fixed array of wireless sensor network nodes equipped with Ethernet interface backchannel boards allowing remote reprogramming and data logging.

- **MoteLab job:** consists of some number of executables and testbed nodes, a description mapping of each node used to an executable, several Java class files used for data logging, and other configuration parameters, such as whether or not to perform power profiling during the experiment.

- **MySQL Database:** there are two kinds of information stored in the database: job-generated data and testbed state. There is a table for each user account, and a set of tables is created for each instance of a job run, one table for each message type associated with the job. A separated database has all lab information, including user information and access rights; node state; information about uploaded executables and class files; job properties; and a representation of the lab schedule.

- **Web Interface:** using PHP to generate dynamic web content, and Javascript to provide an interactive user experience, allowing the user to have access the lab in a platform-independent way.

- **DBLogger:** at the beginning of each job it is started a Java program that connects to each node and uses class to send messages over the serial port and insert in a appropriate database. The individual fields of each sent message are extracted and stored in the database.

- **Job Daemon:** it is responsible to sets up experiments, which include nodes reprogramming, starting other necessary system components, and shutting down when the experiment is finished.

MoteLab provides an experimentation environment similar to most deployments. Its web interface and preemptive scheduler allow a large community to share access to the lab and eliminates the difficulties inherent in cooperative scheduling. It is freely distributed and built on top of readily-available software tools.

3.3 TWIST: TKN Wireless Indoor Sensor network Testbed

The TKN Wireless Indoor Sensor network Testbed (TWIST) [19] is a testbed architecture for indoor deployment of WSN. It provides basic services as: node configuration, network-wide programming, out-of-band extraction of debug data and gathering of application data, and also introduces several novel features.

Figure 3.4 shows the design of TWIST, have a server that is connected via a Ethernet Backbone to the super nodes, many super nodes are connected to USB hubs, and the WSN nodes are connected to USB hubs using the wired infrastructure. The control station is a host that provides a web-based interface. In the follow is describing each part of the system.

- **Sensor nodes:** TWIST is crucially centered on the use of the USB interface, and it is possible to use any platform having a USB 1.1 interface, e.g. eyesIFX, and Telos mote families. Is also uses TinyOS.

- **Testbed Sockets and USB Cabling:** the socket is nothing more than the point where the USB interface of the sensor node attaches to the USB infrastructure. The sockets have unique identifiers, and their geographical position is known and does not change over time. The socket is connected to the testbed using a combination of passive and active USB cables.

- **USB Hubs:** the USB hubs are the central element of TWIST, giving it one of its most important feature: the individually control of power supply of any sensor node in the testbed.

- **Super Nodes:** to solve the problem of scalability, a distributed solution was chosen. The super nodes need to support a communication technology that does not have the size and cable length limits of the USB standard, and forms the testbed backbone to which the server and control stations can be attached.

- **Server:** at the heart of the server there is the PostgreSQL database that stores information of all devices of the system that include node, super node, hubs, sockets, and the locations of the socket.

- **Control Station:** can be any workstation that is attached to the backbone, though the ability to run Linux eases its integration with the testbed.

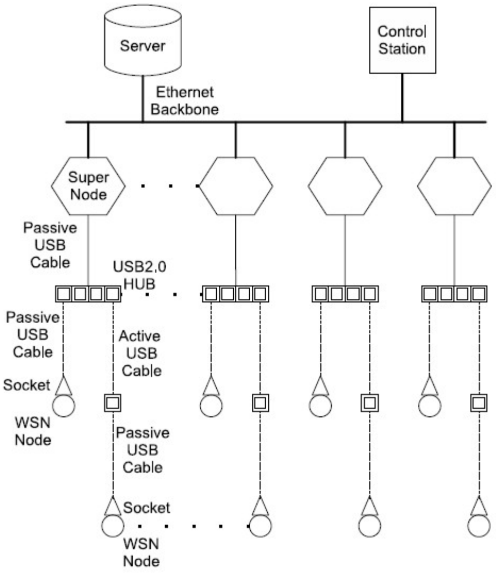


Figure 3.4: Hardware Architecture of TWIST [19]

To perform an experiment in TWIST the user has basically two steps. The first step is select the nodes, the source code to install on the nodes, the channel, and select if is needed start the serial forwarder (this application extract data from the experiment). The second step, consist in select an action that include install the program on the motes, cut or the USB power, and start or stop the serial forwarder.

TWIST supports experiments with heterogeneous node platforms, active power supply control of the nodes (enable switching between USB and battery powered experiments), control the network topology, fast reprogramming, and creation of both flat and hierarchical sensor networks.

3.4 SCALE: A tool for Simple Connectivity Assessment in Lossy Environments

SCALE [9] is a measurement tool to study wireless communication for low power radios. This tool easies the characterization of the links using packet delivery as the basic application-level metric. The tool allows the collection of packet delivery statistics using the same specific hardware platform, and in the same environment. The data gathered by SCALE have interesting implications in the design, evaluation, and parameter tuning of sensor network protocols and algorithms.

The system consists of a number of sensor nodes (Mica 1 and Mica 2) attached to a laptop PC using serial cables to one or more serial multiplexors. The PC performs the data collection of the experiment, and also have a visualization tool integrated that allows visualize the data from experiment in real time, and also the analysis and displays the final experimental results. The sensor nodes firmware comes with an event-driven operating system called TinyOS [20].

SCALE is built using the programming model and software framework named EmStar [14]. Figure 3.5 shows the diagram of the software architecture. The tool is completely modularized, all its modules were written in C. Each module is represented by a process with its own address space. Each node in the experiment runs a software stack, and there are three modules for each software stack: (i) *Conntest* performing the control coordination among nodes, when start/stop send packet; (ii) L*inkStats* is responsible for maintaining the packet delivery statistics from all neighbors; and (iii) *MoteNic* implements the host-mote protocol to communicate with the radio over the serial port. The collection of processes is managed by *emrun*, which starts the modules in a correct dependency order based on the configuration file. If one of the modules close unexpectedly, the emrun restart the module and the other modules can reconnect to it without loosing state, this process is done in a automatically way. *Connview* is the visualization tool.

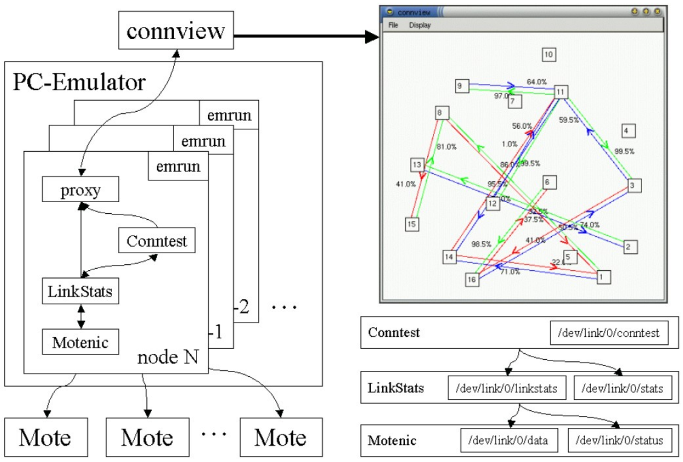


Figure 3.5: SCALE Software Architecture [9]

In SCALE several parameters can be configured in each experiment, such as the number of round-robin passes, the total number of packet probes to be sent (and the number of probes in each round), the packet size, the inter-packet period time, and the transmission power gain. At the end of each experiment, all data is automatically stored in a log files. The log data include: the location of sensor nodes, the collected data during the experiment, the values of all the parameters used, and date/time of the experiment.

3.5 SWAT: Enabling Wireless Network Measurements

The Stanford Wireless Analysis Tool (SWAT) [32] is a software tool that automates gathering and analysis of network measurements. This tool provides an interface for configuring experimental parameters in a network, to gather network data, to distill the data into relevant metrics, and to display the results.

Figure 3.6 shows the structure of the SWAT system. The user specifies the settings of the experiment through the configuration UI, such as: link-layer type (802.15.4/802.11), node list, number of packets, inter-packet interval, type of transmission i.e. broadcast or unicast, CSMA on/off, channel, transmission power, link layer acknowledgements on/off, link layer retransmissions on/off (for unicasts), maximum retransmission count, bit rate (for 802.11), noise sampling on/off, sampling rate, and number of samples.

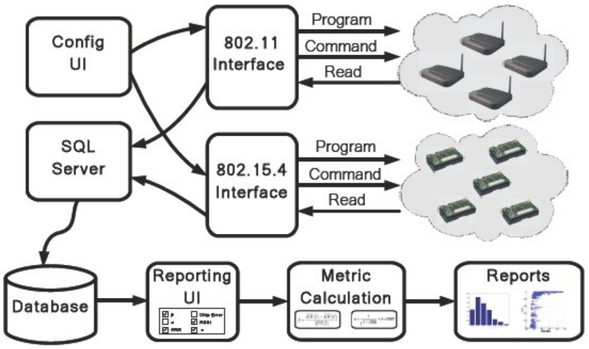


Figure 3.6: Structure of SWAT [32]

The tool uses an appropriate interface written in HTML and PHP to program the nodes, to send command, and to receive packet statistics through a wired or wireless back-channel, and to store the data in a SQL database. Additionally, the researcher can: create reports that consist of the experimental parameters, computed metrics, and generate pertinent reports.

The metrics computed by SWAT include: packet delivery temporal and spatial correlations, noise floor distribution, received signal strength to reception ratio correlation, link asymmetries, and reception ratio over time. Using this toll researchers can comprehend protocol performance in a specific network context, what allows meaningful comparison of protocol performance across different environments.

3.6 Concluding remarks

To evaluate LQEs protocols it is important to have a testbed that allows the collection of all sent and received packets. From each collected packet it is important to extract detailed information, such as: sequence number, sender and receiver identifier (Id), number of retransmissions required before a successful reception, LQI, Signal-to-Noise Ratio (SNR), Received Signal Strength Indicator (RSSI), background noise, and coordinates of the nodes (location).

Additionally, such testbed should also allow nodes configuration to perform an experiment. Possible parameters that can be configured include the channel, transmit power, packet size, number of retransmission, Inter-Packets Interval (IPI), and traffic pattern. Features like nodes reprogramming, automatically detection of connected nodes, easy selection of node to use in the experiment, and off-line analysis are also welcome.

As mentioned along this chapter, several testbeds have been proposed to perform experiments with WSNs in real environment. However, none of these testbeds provide support for all of the mentioned features. Table 3.1 summarizes the expected features, informing about their support in each presented testbeds.

Among these testbeds, only SCALE and SWAT have been devoted for link quality measurements. SCALE uses Mica2 mote and this mote is not suitable for the evaluation of all LQEs, as it does not allow the computation of the LQI metric. Additionally, using SCALE it is not possible to select all experiment parameters mentioned in this sub-section. On the other hand, using SWAT there are some tasks that must be performed manually, especially the detection of the connected motes and its selection for the experiment.

Using TWIST the user is only able to select the channel for the experiment. However, there are more parameters that should be selected for the experiment. This testbed does not detect automatically the connected notes, but does have information about the device (nodes, hubs, and other) stored in a database. Based in this information the user can easily choose the motes for the experiment. Also TWIST does not have tables to store the transmit-

Table 3.1: Comparison Between Testbeds

Features	Mirage	MoteLab	SCALE	SWAT	TWIST
Insert the collected packets into a database	No support	Supported	Supported	Supported	No support
Extract all mentioned data from each packet	Supported	Supported	Partially supported	Supported	Supported
Set up network parameters for the experiment	No support	No support	Partially supported	Supported	Only channel
Nodes reprogramming	Supported	Supported	No support	Supported	Supported
Automatically detection of nodes and easy way to select it	Supported	Supported	Supported	No support	No support detection, but easy to choose nodes
Off-line analysis, create graphs and show data	No support	Support	Support	Support	No support

ted packets during the experiment, therefore it is not possible to perform the off-line analysis.

Mirage and MoteLab are a web-based sensor network testbed, using Mirage and MoteLab it is not possible for the user to set the network parameters for the experiments. Additionally, Mirage does not has tables to store the transmitted data during the experiment to perform off-line analysis.

Chapter 4

The RadiaLE Testbed

In this chapter it is presented the proposed RadiaLE testbed, which aim is to overcome the problems identified in the existing testbeds to perform LQEs evaluation, as discussed in the previous chapter. This chapter details the hardware components of the testbed and the software applications used to control the experiments and collect/analyze the sensors data.

4.1 Overview

RadiaLE framework is developed in collaboration with CISTER Research Group at ISEP with aim to evaluate the performance of LQEs. RadiaLE framework is composed by RadiaLE testbed and DataAnlApp (Data Analysis Application) software tool and RadiaLE testbed includes a suggested hardware architecture and ExpCtrlApp (Experiment Control Application) software tool, as show figure 4.1.

RadiaLE framework allows to perform LQEs evaluation by analyzing their statistical properties, independently of any external factor, such as collisions (each node transmits its data in an exclusive time slot) and routing (using a single-hop network). While the RadiaLE testbed is used for setting up and controlling the experiments, and automating link measurements gathering through packets-statistics collection.

The suggested hardware architecture (or experimentation platform) of the RadiaLE testbed is composed by a set of TelosB motes connected to a control station, which is used for controlling and collecting data from the motes.

ExpCtrlApp is a GUI allowing for: (i) motes selection, programming, and control; (ii) set the network configuration; and (iii) gather and log the experiment-data into a MySQL database. DataAnlApp is used to analyze the collected data, allowing for LQEs evaluation.

While the Experiment Control Application is one of the main contribution of this master thesis, the Data Analysis Application was developed by our collaborating partners, as part of the master thesis from Maissa B. Jamâa, in Sfax, Tunisia [23]. Follows a more detailed discussion about RadiaLE testbed components.

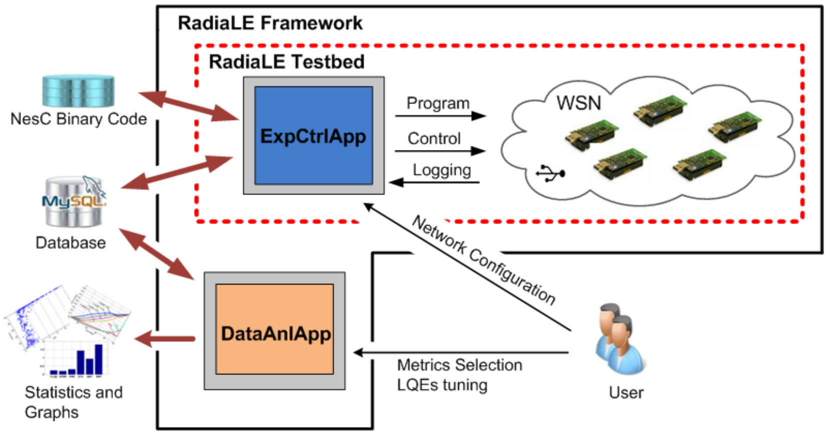


Figure 4.1: Overview of RadiaLE Framework

4.2 Experimentation Platform

The experimentation platform is basically composed by a set of sensor nodes connected to a control station (laptop) through a combination of USB cables and active USB hubs constituting a USB backbone, as shown in figure 4.2. This USB backbone is used to the control station send commands to nodes, receive the packet statistics from nodes, and perform nodes programming. A detailed description of each component of the experimentation platform is given next.

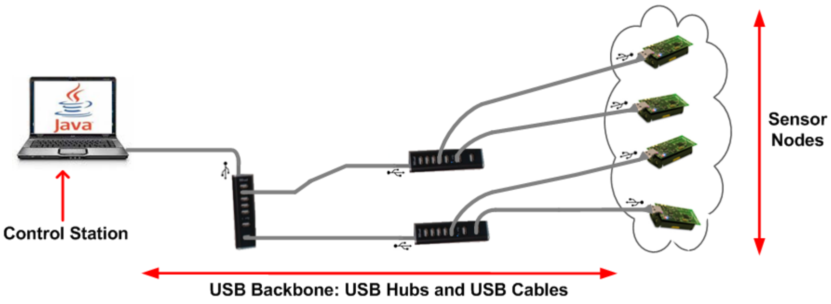


Figure 4.2: Hardware Architecture of RadiaLE Testbed

Sensor Nodes: the sensor nodes must have integrated an USB interface, which provides some interesting features for our testbed: node programming, send commands and receive packet-statistics through the USB backbone. The proposed testbed has support to Tmote sky or TelosB motes. The sensor nodes are programmed in nesC language and the operating system adopted is TinyOS 2.X. TinyOS was chosen because it has support for some interesting features such as, allowing ExpCtrApp to open/close USB connection that are used by the application to send commands and to reprogram the nodes, and the nodes can report about packet-statistics to the control station. Also TinyOS has some implemented functions that allow the motes to get the values of temperature, humidity, light, RSSI, LQI, and background noise. As the sensor nodes are connected to the USB backbone, it is not necessary power supply from battery since it is possible used from USB.

USB Cables and Hubs: USB cables can be passive or active. The main difference between these two kinds of cables is the maximum distance that the cables can transmit data without failures. Passive USB cables can transmit data at maximum distance of 5 meters (m). Active USB cable amplifies the input signal to allow distances of higher than 5m.

Another solution to transmit data over distances higher than 5m is the use of a combination of passive USB cables and active USB Hubs. The active USB hub is an USB multiplexor that connects a set of devices (motes or other USB hubs) and provides power supply. In our testbed we adopted a combination of passive USB cables and active USB Hubs that are used to connect the sensor nodes to the control station.

Control Station: For portability reasons, the control station consists in a laptop that runs the MySQL server and the ExpCtrApp application. The MySQL server is used to log the data from the experiment while the ExpCtrApp application helps the user to perform the experiments and to perform the off-line analysis of data.

4.3 Experiment Control Application

The ExpCtrApp provides a set of facilities for the user to program, control and perform the experiments. It automatically detects the motes connected to the control station, allowing the user to select the motes and to program them. The application also allows users to select a set of network parameters, making it possible to perform experiments under different network configurations. Moreover, during the experiment it allows the user to monitor the collected data and the nodes distribution.

The functionalities of the ExpCtrApp application were organized in two main groups: (i) to perform experiments, and (ii) to make off-line analyzes. Figure 4.3 shows the UML (Unified Modeling Language) use case diagram of the ExpCtrApp application.

To perform an experiment the user should select the motes, program then, and choose the network parameters. During the experiment the user can inspect the collect data and the nodes distribution. All of the experimental parameters and the collect packet-statistics are stored in a MySQL database.

The off-line analysis of data supported by ExpCtrApp offers the user an idea about the collected data during the experiment. To perform the analysis of data, the user should select the desired metrics and can analyze the collected data via graph or simply visualize the raw data as text.

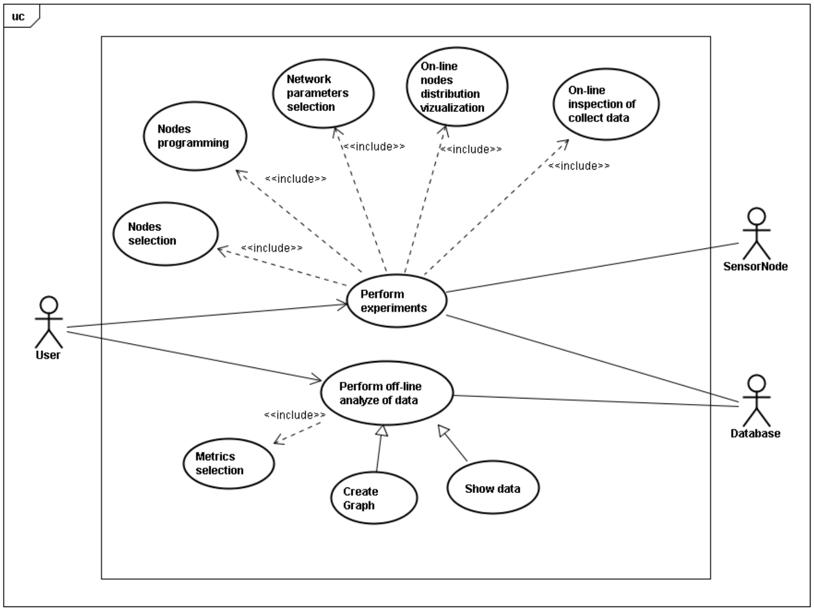

uc
Network parameters selection
On-line nodes distribution vizualization
On-line inspection of collect data
Nodes programming
Nodes selection
<<include>>
Perform experiments
SensorNode
User
Perform off-line analyze of data
Database
Metrics selection
Create Graph
Show data

Figure 4.3: Use Case Diagram

Figure 4.4 presents the activity diagram that depicts the activities related with performing an experiment. Firstly the user must select a set of nodes and program them. After that, to start an experiment, the user needs to fill the fields related to the experimental information and choose a set of network parameters. As the experiment starts, the user should wait for a message informing that the experiment has been finished. While the experiment is running the user can monitor the raw packet-statistics or visualize the nodes distribution. A detailed tutorial explaining how to perform an experiment is presented in the appendix B.

4.3.1 GUI to Perform Experiments

ExpCtrApp provides a set of functionalities to the user to make their experiments. To be more specific, it supports the automatically detection of connected motes, motes programming/control, network configuration, and data logging into a MySQL database, as detailed next. An overview of the created GUI therefore is shown in the figure 4.5.

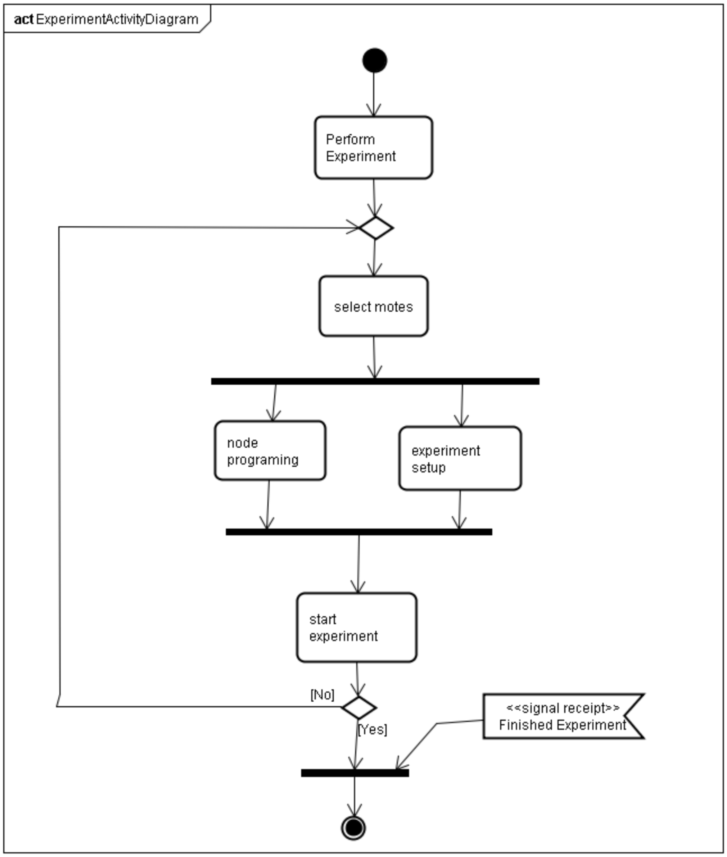
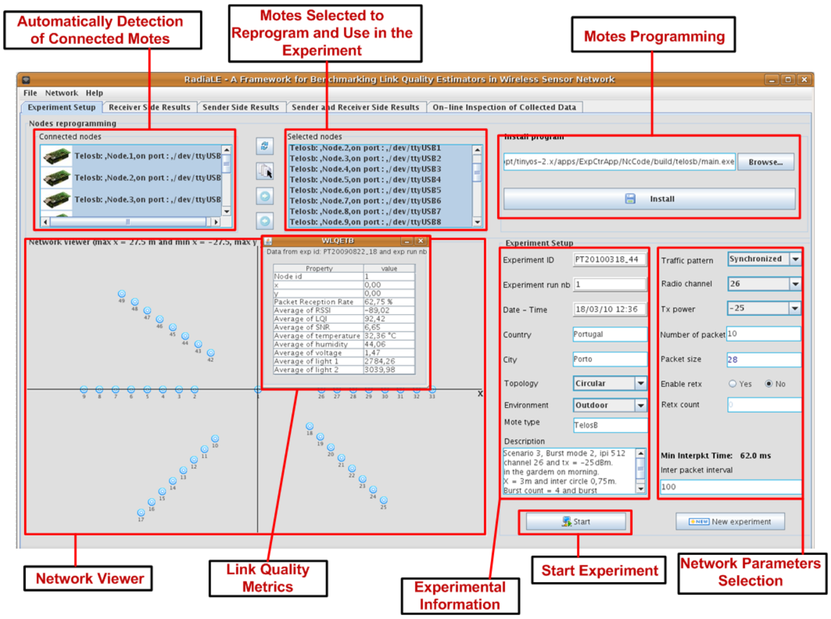


Figure 4.4: Activity Diagram to Perform Experiments

1. **Detection of Connected Motes:** ExpCtrApp automatically detects the sensor nodes connected to the control station and shows to the user a list of connected nodes. The user can select any number of motes to install the binary code, also the selected nodes are used in the experiment. Automatic motes detection is a new functionality that does not exist in other

experimental testbeds and that is very practical, in particular for large-scale experiments. Observe label *Automatically Detection of Connected Motes* at figure 4.5 that displays the connected nodes to the control station.


Automatically Detection of Connected Motes
Motes Selected to Reprogram and Use in the Experiment
Motes Programming
RadiaLE - A Framework for Benchmarking Link Quality Estimators in Wireless Sensor Network
File
Network
Help
Experiment Setup
Receiver Side Results
Sender Side Results
Sender and Receiver Side Results
On-line Inspection of Collected Data
Nodes reprogramming
Connected nodes
Selected nodes
Install program
Browse...
Install
Network viewer (max x = 27.5 m and min x = -27.5, max y
WLQETB
Experiment Setup
Start
New experiment
Network Viewer
Link Quality Metrics
Experimental Information
Start Experiment
Network Parameters Selection

Connected nodes:

Telosb: ,Node.1,on port : ,/dev/ttyUSB
Telosb: ,Node.2,on port : ,/dev/ttyUSB
Telosb: ,Node.3,on port : ,/dev/ttyUSB

Selected nodes:

Telosb: ,Node.2,on port : ,/dev/ttyUSB1
Telosb: ,Node.3,on port : ,/dev/ttyUSB2
Telosb: ,Node.4,on port : ,/dev/ttyUSB3
Telosb: ,Node.5,on port : ,/dev/ttyUSB4
Telosb: ,Node.6,on port : ,/dev/ttyUSB5
Telosb: ,Node.7,on port : ,/dev/ttyUSB6
Telosb: ,Node.8,on port : ,/dev/ttyUSB7
Telosb: ,Node.9,on port : ,/dev/ttyUSB8

Install program: pt/tinyos-2.x/apps/ExpCtrApp/NcCode/build/telosb/main.exe

Data from exp id: PT20090822_18 and exp run nb

Property	value
Node id	1
x	0,00
y	0,00
Packet Reception Rate	62,75 %
Average of RSSI	-89,02
Average of LQI	92,42
Average of SNR	6,65
Average of temperature	32,36 °C
Average of humidity	44,06
Average of voltage	1,47
Average of light 1	2784,26
Average of light 2	3039,98

Experiment ID	PT20100318_44
Experiment run nb	1
Date - Time	18/03/10 12:36
Country	Portugal
City	Porto
Topology	Circular
Environment	Outdoor
Mote type	TelosB

Description: Scenario 3, Burst mode 2, ipi 512 channel 26 and tx = -25dBm. in the gardem on morning. X = 3m and inter circle 0,75m. Burst count = 4 and burst

Traffic pattern	Synchronized
Radio channel	26
Tx power	-25
Number of packet	10
Packet size	28
Enable retx	Yes [✓ No]
Retx count	0

Min Interpkt Time: 62.0 ms

Inter packet interval: 100

Figure 4.5: ExpCtrApp Interface to Perform Experiments

2. **Motes Programming:** ExpCtrApp automatically detects the motes connected to the Laptop, after that the user select the nodes and the application uploads the nesC program binary code. This functionality is very important for experiments where the user should reprogram the nodes several times. See label *Motes Programing* from figure 4.5 that shows the interface to programming the selected node.

3. **Network Configuration:** The application enables the user to select the set of network parameters that will be used for the motes during the experiment. Possible parameters that can be configured include the radio channel, transmission power, packet size, number of sent packets,

IPI, traffic pattern, retransmission on/off, and the maximum number of retransmissions. As the user starts the experiment, using the USB backbone the application sends a configuration message to the nodes informing the network parameters that will be used in the experiment. observe the label *Network Parameters Selection* at figure 4.5, that depict the interface to select the network parameters.

4. **Motes Control:** ExpCtrApp sends commands and receive reports from motes to control data transmission according to the selected traffic pattern in the network configuration step. More details about the supported traffic patterns are presented latter in this subsection.

5. **Link Measurements Collection:** The nodes exchange data and collect packet-statistics from received packets: sequence number, sender and receiver Id, number of retransmissions required before a successful reception, LQI, SNR, RSSI, background noise, timestamp, motes coordinates, temperature, humidity, and light. Other data are collected at the sender side, such as packet sequence number, retransmission count, and sender and destination Id. All these data from sender and receiver side, are forwarded via the USB backbone to the ExpCtrApp in the control station and then stored in a database.

4.3.2 Supported Traffic Patterns

To accurately assess the link asymmetry property, related with LQE evaluation it is necessary to collect packet-statistics on both link directions at (almost) the same time. Most of the existing testbeds use one-burst traffic, where each node sends a burst of packets to each of their neighbors then passes the token to the next node to send its burst. This traffic pattern cannot accurately capture the link asymmetry property as the two directions (uplink and downlink) will be assessed in separate time windows. This traffic pattern is definitely inappropriate for the assessment of link asymmetry. The synchronized traffic pattern is more convenient than the burst traffic (in particular for large bursts) to evaluate link asymmetry, which must be evaluated in short time windows.

Thus, using different traffic patterns that improve the accuracy of link asymmetry assessment is mandatory. For that reason, the proposed testbed has support for both burst and synchronized traffic patterns. This is important allow understanding the performance of LQEs for different traffic configurations/patterns. Another reason for supporting two traffic patterns is that the radio channels exhibit different behaviors with respect to these two traffic patterns, as further discussed.

Figures 4.6 and 4.7 illustrate both implemented traffic patterns. More specifically, these figures shows the interaction between the PC and two nodes constituting the link Mote$_1$ $\longleftrightarrow$ Mote$_i$. For the Burst traffic, given a couple of motes (Mote$_1$ and Mote$_i$), the Mote$_1$ sends a burst packets to Mote$_i$, the burst packets is composed by N packets defined by the user. When it finishes, it sends a command reporting finish of transmission to the control station, allowing Mote $_i$ to send its burst of packets to Mote$_1$. When Mote$_1$ finishes the transmission, it acknowledges the control station. This operation is repeated for a certain number of bursts defined for the user. When these motes finish sending packets, the next couple of motes will repeat this process, (Mote$_1$ and Mote$_{i+1}$).

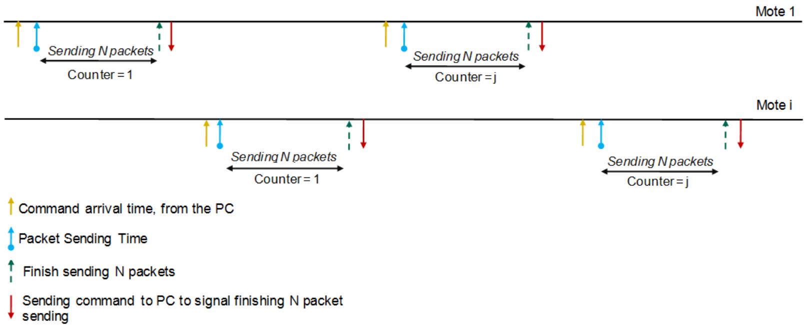


Figure 4.6: Burst Traffic

For the synchronized traffic, see figure 4.7, a couple of motes (Mote $_1$ and Mote $_i$) are exchanging packets in a synchronized way, one packet at each time. Follow the a description of the behavior of the synchronized traffic, the control station sends a command to Mote$_1$ and Mote$_i$ indicating the beginning of transmission, and the couple of motes start sending packets using an exclusive time slot (to avoid collisions). When this couple of motes finish, they send a command reporting finish of transmission to the control station, and the control station sends a command to the next couple of motes (mote$_1$ and Mote$_{i+1}$).

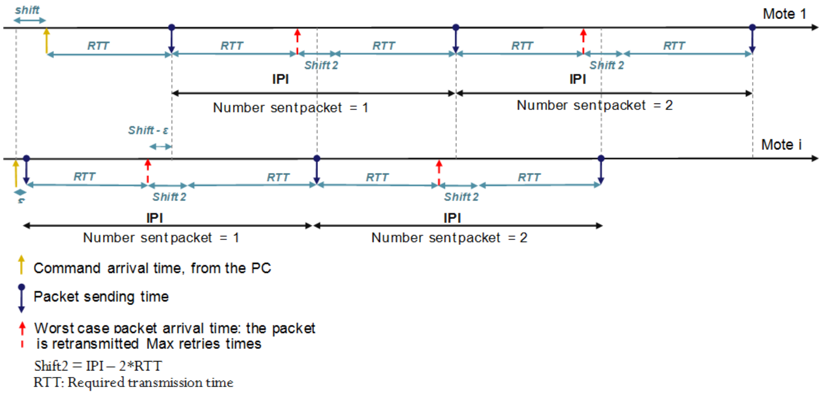


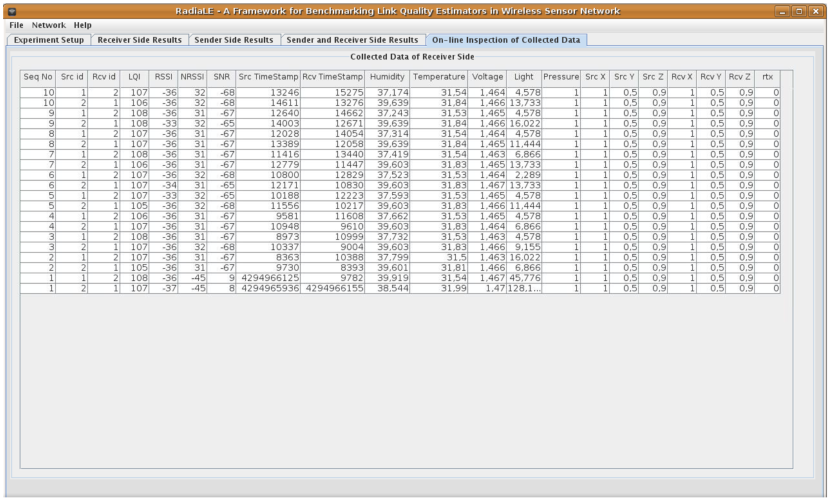
Figure 4.7: Synchronized Traffic

4.3.3 On-line Analysis

The developed software tool also includes the network viewer (observe the label *Network Viewer* at figure 4.5) that displays on-line the network distribution. To draw the nodes distribution the application uses the location in Cartesian coordinates (x, y) that are added in the packet-statistics of the receiver packet. Each mote stores the location coordinates in its source code, which was manually inserted by the user.

If the user click on the icon that represent the node on network viewer, it will open a new interface showing some link quality metrics in a table (e.g. PRR, RSSI, and others), see label *Link Quality Metrics* from figure 4.5.

Additionally the user can monitor the raw data received from the nodes in real-time, as shows figure 4.8. This interface is constituted by a table that shows the packet-statistics from receiver side as text.


RadiaLE - A Framework for Benchmarking Link Quality Estimators in Wireless Sensor Network
File
Network
Help
Experiment Setup
Receiver Side Results
Sender Side Results
Sender and Receiver Side Results
On-line Inspection of Collected Data

Collected Data of Receiver Side

Seq No	Src id	Rcv id	LQI	RSSI	NRSSI	SNR	Src TimeStamp	Rcv TimeStamp	Humidity	Temperature	Voltage	Light	Pressure	Src X	Src Y	Src Z	Rcv X	Rcv Y	Rcv Z	rtx
10	1	2	107	-36	32	-68	13246	15275	37,174	31,54	1,464	4,578	1	1	0,5	0,9	1	0,5	0,9	0
10	2	1	106	-36	32	-68	14611	13276	39,639	31,84	1,466	13,733	1	1	0,5	0,9	1	0,5	0,9	0
9	1	2	108	-36	31	-67	12640	14662	37,243	31,53	1,465	4,578	1	1	0,5	0,9	1	0,5	0,9	0
9	2	1	108	-33	32	-65	14003	12671	39,639	31,84	1,466	16,022	1	1	0,5	0,9	1	0,5	0,9	0
8	1	2	107	-36	31	-67	12028	14054	37,314	31,54	1,464	4,578	1	1	0,5	0,9	1	0,5	0,9	0
8	2	1	107	-36	31	-67	13389	12058	39,639	31,84	1,465	11,444	1	1	0,5	0,9	1	0,5	0,9	0
7	1	2	108	-36	31	-67	11416	13440	37,419	31,54	1,463	6,866	1	1	0,5	0,9	1	0,5	0,9	0
7	2	1	106	-36	31	-67	12779	11447	39,603	31,83	1,465	13,733	1	1	0,5	0,9	1	0,5	0,9	0
6	1	2	107	-36	32	-68	10800	12829	37,523	31,53	1,464	2,289	1	1	0,5	0,9	1	0,5	0,9	0
6	2	1	107	-34	31	-65	12171	10830	39,603	31,83	1,467	13,733	1	1	0,5	0,9	1	0,5	0,9	0
5	1	2	107	-33	32	-65	10188	12223	37,593	31,53	1,465	4,578	1	1	0,5	0,9	1	0,5	0,9	0
5	2	1	105	-36	32	-68	11556	10217	39,603	31,83	1,466	11,444	1	1	0,5	0,9	1	0,5	0,9	0
4	1	2	106	-36	31	-67	9581	11608	37,662	31,53	1,465	4,578	1	1	0,5	0,9	1	0,5	0,9	0
4	2	1	107	-36	31	-67	10948	9610	39,603	31,83	1,464	6,866	1	1	0,5	0,9	1	0,5	0,9	0
3	1	2	108	-36	31	-67	8973	10999	37,732	31,53	1,463	4,578	1	1	0,5	0,9	1	0,5	0,9	0
3	2	1	107	-36	32	-68	10337	9004	39,603	31,83	1,466	9,155	1	1	0,5	0,9	1	0,5	0,9	0
2	1	2	107	-36	31	-67	8363	10388	37,799	31,5	1,463	16,022	1	1	0,5	0,9	1	0,5	0,9	0
2	2	1	105	-36	31	-67	9730	8393	39,601	31,81	1,466	6,866	1	1	0,5	0,9	1	0,5	0,9	0
1	1	2	108	-36	-45	9	4294966125	9782	39,919	31,54	1,467	45,776	1	1	0,5	0,9	1	0,5	0,9	0
1	2	1	107	-37	-45	8	4294965936	4294966155	38,544	31,99	1,47	128,1...	1	1	0,5	0,9	1	0,5	0,9	0

Figure 4.8: ExpCtrApp Interface for On-line Visualization of the Collected Data

4.3.4 Off-line Data Analysis

As was already mentioned, the off-line data analysis supported by ExpCtrApp serves just to give the user an idea about the collected data. A thorough analysis of data is provided by the DataAnlApp application [22].

The ExpCtrApp application stores in a database the packet-statistics of the sender and receiver side, then the application provides three interfaces to analyze the results: analyze the received side results, the sender side results, and both the sender and received side results. These three kinds of visualization have a similar sequence of activities, as it can be observed in Figure 4.9.

Let us consider the case where the user selects to obtain the receiver side results. First, the user should select four parameters: (i) experiment identifier; (ii) experiment run number; (iii) sender Id; (iv) receiver Id. After that the user can import the data and visualize it either as a simple table or in the graphic format. If the user imports the data or creates a graph, the experimental information are displayed to help the user to be aware about the setup used on the selected experiment. To delete the information related to the selected experiment, the user just needs to select the experiment Id and the experiment run number. A tutorial explaining how to perform the off-line analysis is presented in the appendix C.

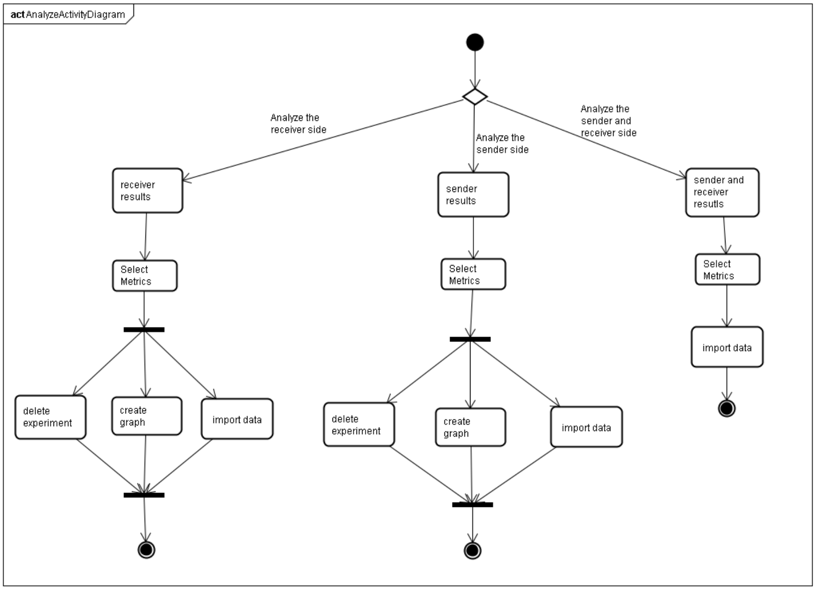
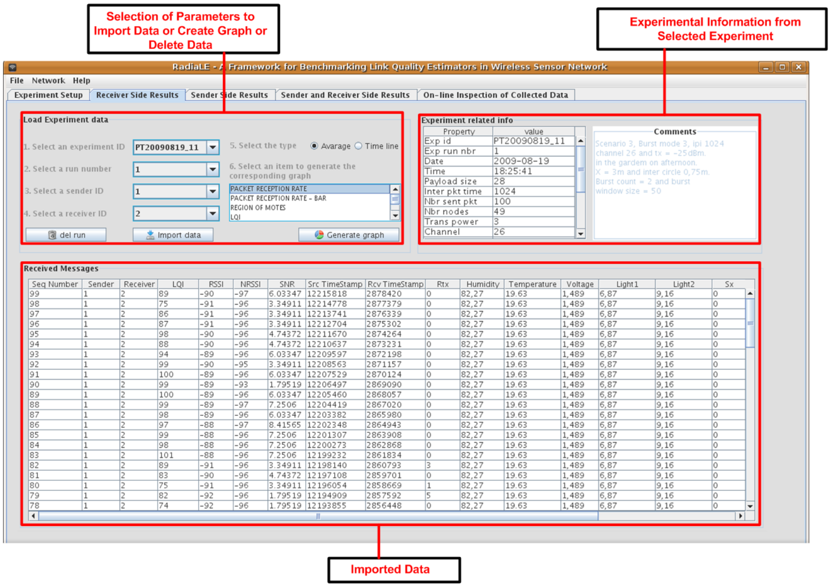

act AnalyzeActivityDiagram
Analyze the receiver side
Analyze the sender side
Analyze the sender and receiver side
receiver results
sender results
sender and receiver resutls
Select Metrics
delete experiment
create graph
import data

Figure 4.9: Activity Diagram to Perform Off-line Analyze of Data

Figure 4.10 presents an overview of the interface to analyze the receiver side results. The user can visualize the collected data in a table or create some graphs. The user can create multiple graphs by selecting some metrics, such as PRR, region of connectivity of nodes (connected, disconnected or transitional), LQI, RSSI, background noise, SNR, humidity, temperature, light, and the nodes distribution. The graph can show the selected metric by average or time line, except for PRR and region of connectivity of nodes, which are only available as average graph.

Some useful information that describes the experiment are automatically stored in a database, when the user starts the experiment. This information includes the selected network parameters, date and time, topology used, country and city, experiment Id, environment (indoor or outdoor), motes type, and a brief description of the experiment. These experiment-related information are very important to register the settings used for each experiment during the data analysis.

Both for creating graphs and to visualize data, the user can see the information related in the experiment, such as showed in the label *Experimental information from imported experiment* from figure 4.10.


Selection of Parameters to Import Data or Create Graph or Delete Data
Experimental Information from Selected Experiment
Imported Data
RadiaLE - A Framework for Benchmarking Link Quality Estimators in Wireless Sensor Network
File
Network
Help
Experiment Setup
Receiver Side Results
Sender Side Results
Sender and Receiver Side Results
On-line Inspection of Collected Data
del run
Import data
Generate graph

Load Experiment data

1. Select an experiment ID: PT20090819_11
2. Select a run number: 1
3. Select a sender ID: 1
4. Select a receiver ID: 2
5. Select the type: [✓ Avarage] Time line
6. Select an item to generate the corresponding graph: [✓ PACKET RECEPTION RATE] PACKET RECEPTION RATE - BAR, REGION OF MOTES, LQI

Experiment related info

Property	value
Exp id	PT20090819_11
Exp run nbr	1
Date	2009-08-19
Time	18:25:41
Payload size	28
Inter pkt time	1024
Nbr sent pkt	100
Nbr nodes	49
Trans power	3
Channel	26

Comments

Scenario 3, Burst mode 3, ipi 1024
channel 26 and tx = -25dBm.
in the gardem on afternoon.
X = 3m and inter circle 0,75m.
Burst count = 2 and burst
window size = 50

Received Messages

Seq Number	Sender	Receiver	LQI	RSSI	NRSSI	SNR	Src TimeStamp	Rcv TimeStamp	Rtx	Humidity	Temperature	Voltage	Light1	Light2	Sx
99	1	2	89	-90	-97	6.03347	12215818	2878420	0	82,27	19.63	1,489	6,87	9,16	0
98	1	2	75	-91	-96	3.34911	12214778	2877379	0	82,27	19.63	1,489	6,87	9,16	0
97	1	2	86	-91	-96	3.34911	12213741	2876339	0	82,27	19.63	1,489	6,87	9,16	0
96	1	2	87	-91	-96	3.34911	12212704	2875302	0	82,27	19.63	1,489	6,87	9,16	0
95	1	2	98	-90	-96	4.74372	12211670	2874264	0	82,27	19.63	1,489	6,87	9,16	0
94	1	2	88	-90	-96	4.74372	12210637	2873231	0	82,27	19.63	1,489	6,87	9,16	0
93	1	2	94	-89	-96	6.03347	12209597	2872198	0	82,27	19.63	1,489	6,87	9,16	0
92	1	2	99	-90	-95	3.34911	12208563	2871157	0	82,27	19.63	1,489	6,87	9,16	0
91	1	2	100	-89	-96	6.03347	12207529	2870124	0	82,27	19.63	1,489	6,87	9,16	0
90	1	2	99	-89	-93	1.79519	12206497	2869090	0	82,27	19.63	1,489	6,87	9,16	0
89	1	2	100	-89	-96	6.03347	12205460	2868057	0	82,27	19.63	1,489	6,87	9,16	0
88	1	2	99	-89	-97	7.2506	12204419	2867020	0	82,27	19.63	1,489	6,87	9,16	0
87	1	2	98	-89	-96	6.03347	12203382	2865980	0	82,27	19.63	1,489	6,87	9,16	0
86	1	2	97	-88	-97	8.41565	12202348	2864943	0	82,27	19.63	1,489	6,87	9,16	0
85	1	2	99	-88	-96	7.2506	12201307	2863908	0	82,27	19.63	1,489	6,87	9,16	0
84	1	2	98	-88	-96	7.2506	12200273	2862868	0	82,27	19.63	1,489	6,87	9,16	0
83	1	2	101	-88	-96	7.2506	12199232	2861834	0	82,27	19.63	1,489	6,87	9,16	0
82	1	2	89	-91	-96	3.34911	12198140	2860793	3	82,27	19.63	1,489	6,87	9,16	0
81	1	2	83	-90	-96	4.74372	12197108	2859701	0	82,27	19.63	1,489	6,87	9,16	0
80	1	2	75	-91	-96	3.34911	12196054	2858669	1	82,27	19.63	1,489	6,87	9,16	0
79	1	2	82	-92	-96	1.79519	12194909	2857592	5	82,27	19.63	1,489	6,87	9,16	0
78	1	2	74	-92	-96	1.79519	12193855	2856448	0	82,27	19.63	1,489	6,87	9,16	0

Figure 4.10: ExpCtrApp Interface to Perform the Off-line Analysis of Data

4.4 Project Design

This section presents in details the software-design of the ExpCtrApp. This application was developed in the Java language to easy its portability.

4.4.1 Class Diagram

Figure 4.11 shows its class diagram, where attributes and methods are omitted to allow a better visualization of the diagram. Follows a detailed explanation of the main classes that constitute the system.

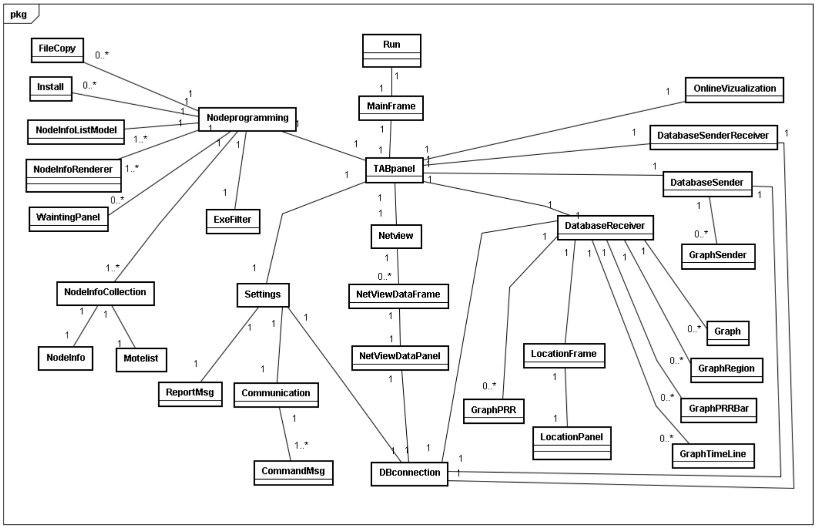


Figure 4.11: Class Diagram

The class **Settings** implements the user interface to start the experiment, to select the network parameters, and to fill the experimental information. This class has some methods to control the experiment by sending command messages to nodes to start exchanging data traffic and receive the packet statistic from motes. **ReportMsg** has methods used by the application to get the values from a received packet of a mote. To send a message to nodes, the application uses the class **CommandMsg**.

To open and close the connection with the database it is used the class **DBConnection**, which also allows to select, insert, delete, and update data from a selected table.

The class **NodeProgramming** has implemented the interface that shows the list of connected nodes and enables the user to select the nodes and program them. The class **Motelist** returns the list of connected nodes to control station. To upload the binary code on sensor nodes it is used the class **Install**.

Netview is the interface used to display the nodes distribution, this class has methods to paint and delete the nodes on interface.

DatabaseSender, **DatabaseReceiver**, **DatabaseSenderReceiver** are the classes that implement the interfaces to analyze the results. These classes have methods that allow importing, deleting, or creating graphs according to the selected metrics. These classes use one of the classes that have the word graph in the name of the class to create graphs of timeline or average.

To show the collected data during the experiment it is used the class **OnlineVizualization** that implements the interface used to show the collected data.

4.4.2 Database Project

The database project consists of three tables: **linkData**, **experiment**, and **SenderSide**, as depicted in the Entity-Relationship Diagram of figure 4.12. As it can be observed, the table **experiment** can have *n* data from the table **linkData** or **SenderSide**.

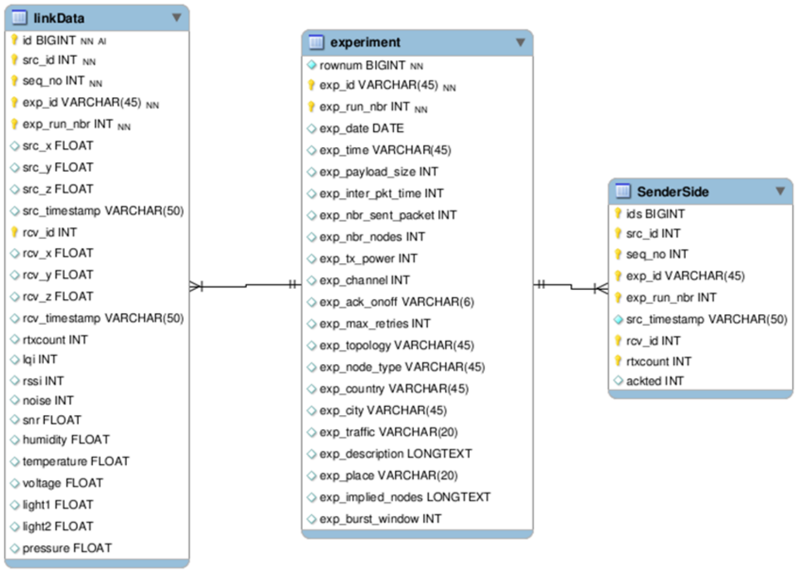


Figure 4.12: Entity-Relationship Diagram

The table **experiment** stores information related to the experiments, which includes the selected network parameters and the experimental information. Information related to the experiment includes date, time, city, country, experiment Id, topology, kind of mote, environment type, and a briefed description. The network parameters include IPI, radio channel, transmission power, packet size, number of sent packets, number of used node, and selected traffic pattern.

The **linkData** table contains information about packet-statistics of the receiver side, such as sender and receiver Id, location coordinates from sender and receiver, timestamp from sender and receiver, number of retransmission, LQI, RSSI, background noise, and SNR. Also, information of the temperature, humidity, pressure and lights are stored in a database.

The packet-statistics of the sent packets are stored in the table **Sender-Side**. This information includes the sequence number, source Id, destination Id, timestamp, number of retransmission of each sent packet, and if the packet receive the acknowledgment or not.

4.4.3 Behavior Modeling

UML sequence diagrams are used to illustrate the message exchange (i.e. method calls) between several objects in a specific time-delimited situation. Figure 4.13 shows the sequence diagram to install the source code on the sensor nodes.

To program a node the user clicks the button install in the interface of **NodeProgramming**, following it is called the method *nodeProgram* in the class **Install** and this method receives the arguments *id of nodes* and *comport* that the node is connected and then it is installed the nesC source code on **SensorNode**. It shows an interface to the user wait to finish the install process, the class **Install** calls the method *initialize* in the class **WaintingPanel** to show the interface for the user wait.

Figure 4.14 shows the sequence diagram to perform an experiment. User clicks on the button start on interface **Setting** and then it is called the method *InsertExperiment* in the class **DBConnection** to insert in the

Database the data on table experiment. After this the class **Setting** calls the method *sendPackets* to send a command message to **SensorNode** informing the network parameters and to start exchange data traffic.

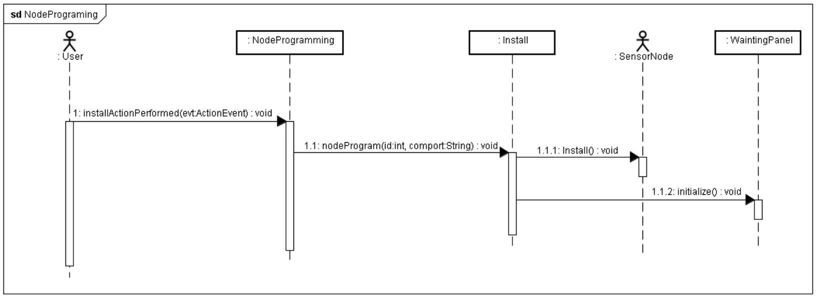

sd NodePrograming
: User
: NodeProgramming
: Install
: SensorNode
: WaintingPanel
1: installActionPerformed(evt:ActionEvent) : void
1.1: nodeProgram(id:int, comport:String) : void
1.1.1: Install() : void
1.1.2: initialize() : void

Figure 4.13: Sequence Diagram to Install the Source Code on Sensor Nodes

Method *messageReceived* receives the packets-statistics from **SensorNodes** via USB backbone and for each received packet, if the packet-statistics is from the sender side it is called the method *insertSender* on class **DBConnection** to insert in the **Database** the data on table linkData. Otherwise it is called the method *insertLinkdata* on **DBConnection** to insert data on table linkData after that the method paint is called to draw the node on interface **Netview**, finally it is called the method *setTable* to show the received packet on the interface of the class **OnlineVizualization**.

To select the metrics for off-line analyzes follow a figure 4.15 showing the sequence diagram. First the user selects the experiment Id on interface of the class **DatabaseReceiver**, then this class call the method select that receive the argument sql, to select the list of experiment run number of the selected experiment Id.

The process is repeated as the user select the experiment run number and the class **DatabaseReceiver** select the list of sender, also when the user select the sender the class select the list of receiver according to the selected parameters.

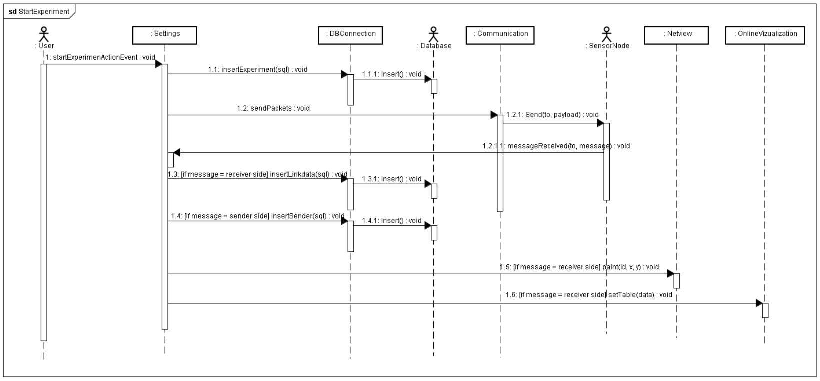

sd StartExperiment
: User
: Settings
: DBConnection
: Database
: Communication
: SensorNode
: Netview
: OnlineVizualization
1: startExperimenActionEvent : void
1.1: insertExperiment(sql) : void
1.1.1: Insert() : void
1.2: sendPackets : void
1.2.1: Send(to, payload) : void
1.2.1.1: messageReceived(to, message) : void
1.3: [if message = receiver side] insertLinkdata(sql) : void
1.3.1: Insert() : void
1.4: [if message = sender side] insertSender(sql) : void
1.4.1: Insert() : void
1.5: [if message = receiver side] paint(id, x, y) : void
1.6: [if message = receiver side] setTable(data) : void

Figure 4.14: Sequence Diagram to Perform an Experiment

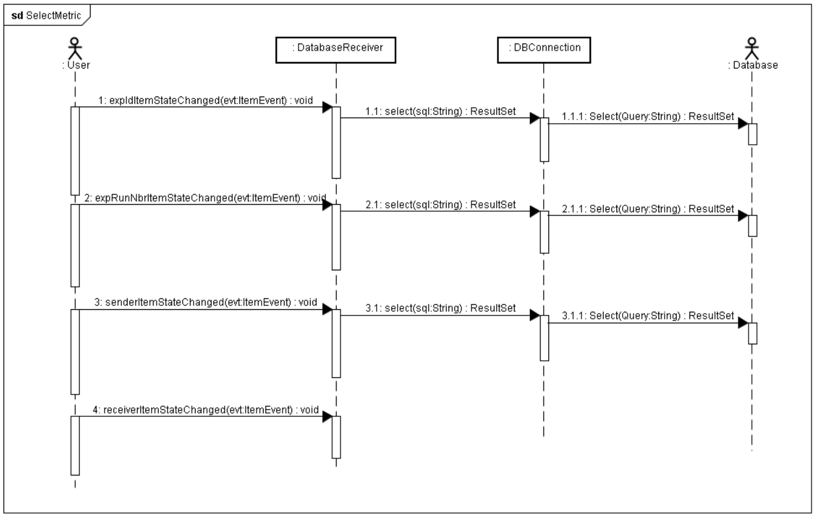

sd SelectMetric
: User
: DatabaseReceiver
: DBConnection
: Database
1: expIdItemStateChanged(evt:ItemEvent) : void
1.1: select(sql:String) : ResultSet
1.1.1: Select(Query:String) : ResultSet
2: expRunNbrItemStateChanged(evt:ItemEvent) : void
2.1: select(sql:String) : ResultSet
2.1.1: Select(Query:String) : ResultSet
3: senderItemStateChanged(evt:ItemEvent) : void
3.1: select(sql:String) : ResultSet
3.1.1: Select(Query:String) : ResultSet
4: receiverItemStateChanged(evt:ItemEvent) : void

Figure 4.15: Sequence Diagram to Perform the Off-line Analyze

4.5 Limitations of RadiaLE Testbed

An USB network is composed by USB hubs and cables. Cables can be maximum 5m long and the hubs can be cascaded at most 5 times. This implies that the adopted USB network can achieve the maximum distance of 30 m, and is limited to 127 devices.

Another limitation is that RadiaLE is built based on the use of TelosB or Tmote sky. As mentioned earlier, this restriction comes from the fact that these motes already incorporate an USB interface, and also provide some important features, such as: list the connected nodes, allow the nodes programming, send command and receive packet-statistics.

Finally, RadiaLE relies on the use of TinyOS because it implements some useful functions that allows to: open and close USB connection between control station and sensor nodes, and to the sensor nodes get the values of temperature, humidity, light, RSSI, LQI, and background noise.

4.6 Concluding remarks

In this chapter it was shown the main functionalities of the RadiaLE testbed including a description the adopted hardware architecture, its deployment, and the developed software tool. The proposed testbed helps the user to perform experiments in WSN using different network setting, to select and reprogram the motes and to gather and log the packet-statistics. RadiaLE overcomes the existing Testbeds given its large set of functions, as summarized in Table 4.1.

RadiaLE is a result of a collaborative work, and is now left as open source for the community (see [38]), and a detailed tutorial explaining how to install the RadiaLE testbed is presented in the appendix A.

Table 4.1: Comparison Between Testbeds, Including RadiaLE Testbed

Features	Mirage	MoteLab	SCALE	SWAT	TWIST	RadiaLE
Insert the collected packets into a database	No support	Supported	Supported	Supported	No support	Supported
Extract all mentioned data from each packet	Supported	Supported	Partially supported	Supported	Supported	Supported
Set up network parameters for the experiment	No support	No support	Partially supported	Supported	Only channel	Supported
Nodes programming	Supported	Supported	No support	Supported	Supported	Supported
Automatically detection of nodes and easy way to select it	Supported	Supported	Supported	No support	No detect nodes but easy to select it	Supported
create graphs and show data	No support	Support	Support	Support	No support	Supported

Chapter 5

Case Study: LQEs Performance Evaluation

The previous chapter presented the details and the main features of the RadiaLE testbed. To illustrate the usefulness of the proposed testbed, in this chapter is presented a case study that uses RadiaLE testbed to make the performance evaluation of several LQEs. By the way, this study can be seen as the primary goal of the proposed testbed. Obtaining LQE detailed information is useful to help higher layers protocols designers, making them aware of those LQE that are most resistive to transient fluctuations in the link quality [4].

The performed experiments are a real implementation of the simulation study presented in [4], which consisted of a simulation comparison of several link quality estimators in WSNs. The LQEs under evaluation are: PRR, ETX [13], RNP [10], Four-bit [16], WMEWMA [42], and our proposed F-LQE [7].

The experiments consist of setting-up the testbed, collecting the packet-statistics within different network settings, and perform the evaluation of several LQEs in terms of reliability and stability, as detailed along this chapter.

5.1 Experiment Description

The first step in order to evaluate the performance of LQEs is to establish a rich set of links with different link qualities. The second step it is create a bidirectional data traffic over each link, enabling link measurements through the packet-statistics (such as packet sequence number, from received and sent packets) collection. Finally, the collected data is analyzed, enabling the evaluation of LQEs.

We used the experimentation platform presented in section 4.2 to create a single-hop sensor network containing a set of TelosB motes positioned in a circular topology. External factors such as routing and collision were not considered. Our goal was to compare the stability property of these LQEs for different network settings.

The experiments discussed here were performed in an outdoor environment, in a garden of ISEP/Porto, as shown in figure 5.1. This area has approximately 30 by 28 meters, it is surrounded by buildings and trees, and has an open space in the center. It was necessary to deploy the network topology every day, which implied in reinstalling the source code on the sensor nodes on every experiment.

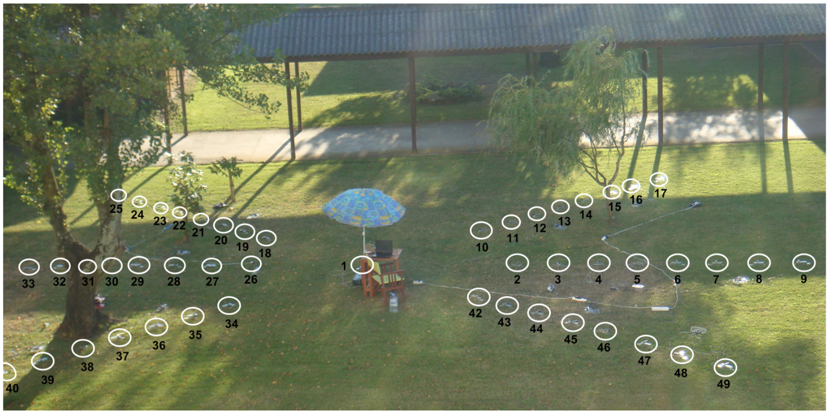


Figure 5.1: Nodes Distribution According the Circular Topology, in a Garden of ISEP

We adopted a circular topology using 49 TelosB motes, N_1 to N_{49}, as shown in figure 5.2. This topology contains a central mote, N_1, while the other 48 motes are divided in 8 set of 6 motes. Each set of motes is placed in a circle around the central node. The first circle, which is nearer to N_1, is distant X meters, and each two consecutive circles are separated by Y. Since the distance and the direction are fundamental factors that affect the link quality, the underlying links $N_1 \leftrightarrow N_n$ will have different characteristics (qualities) by placing nodes N_2 to N_{49} at different distances and directions from the central node N_1. Thus, it is recommended to empirically determine the most appropriate value for X and Y, prior to experiments, to better explore the spatial characteristics of the transitional region, which is typically quantified in the literature by means of the PRR. The transitional region is characterized by a node has the PRR between 10 % and 90 % [43] [10], which means that the links have moderate connectivity.

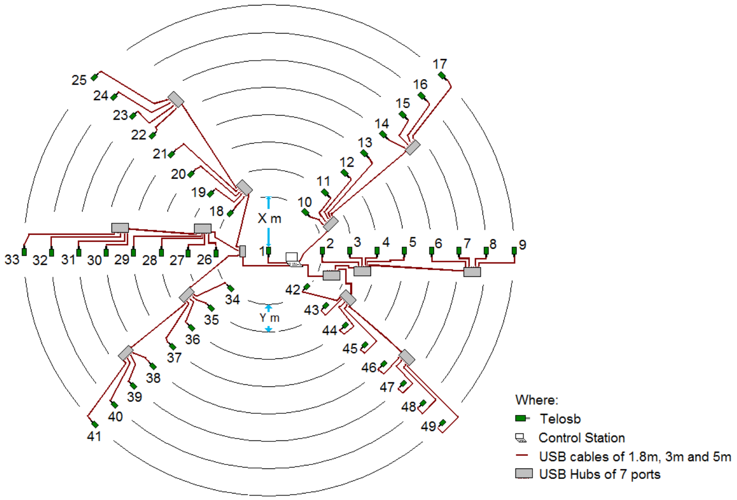


Figure 5.2: Nodes Distributed According the Circular Topology

Several experiments where performed to choose the best X and Y value, which consumed about 7 days with each experiment taking approximately 6 hours to complete. In each experiment, we set transmission power at -25dBm (minimal value), use channel 26, and modified the X and Y value to an arbitrary value. At the end of the experiment, we computed the PRR for each link.

We chose the X and Y value in a convenient way, so that most of the links belong to the transitional region. Figure 5.3 shows the connectives regions (connected, disconnected and transitional).

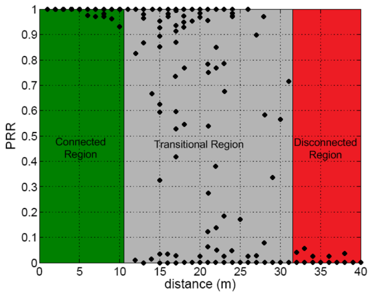


Figure 5.3: Regions of Connectivity

Note that the average PRR of a given link is the average over different PRR samples. Each PRR sample is computed based on W received packets, where W is the estimation window. As we have mentioned before, the transitional region is the most relevant context to assess the performance of LQEs. We concluded from the experiments that using Y of 0.75 meter, and X in a set of $\{2 \text{ and } 3 \text{ meters}\}$ we have the transitional region for most of the links.

As we used a public outdoor environment to make the experiments, it was necessary to deploy the nodes every day. To ensure that the motes were deployed at the same location and has the same identifier in all experiments, we needed to measure the motes coordinates. Each mote has the Cartesian coordinates values in three dimensions (x, y, z). It was considered that the measured coordinates can have an error of approximately 5 cm. Theses coordinates values were used to: (i) deploy the nodes every day in the same location; (ii) for to ExpCtrApp draw the network topology while the experiment is running; (iii) to create graphs and analyzing the results based on the distance among the nodes.

As already mentioned, we did not consider collisions. Therefore, we adopted a collision-avoidance procedure that consisted of guaranteeing that the IEEE 802.15.4 physical channel is free from the interference of IEEE 802.11 networks, which is very common in the university area and operates at the same frequency range. Thus, we adopt as default for our experiment the IEEE 802.15.4 channel 26, which is outside of the IEEE 802.11 frequency range [25].

Several experiments were conducted under different network conditions. In each experiment, we modified a given parameter to study its impact, and the experiment was repeated for each parameter modification. Parameters like traffic type, packet size, radio channel, and maximum number of retransmissions (Rtx) where changed to create 5 different scenarios. The duration of each experiment was approximately 8 hours, and the overall experiments consumed about 15 days. Table 5.1 summarizes the different settings for each experiment.

Table 5.1: Experimental Scenarios. Burst (X, Y, Z) and Synch (W, Z); X: Number of packets per burst, Y: number of bursts, Z: IPI in ms, W: total number of packets.

Scenarios	Traffic Pattern	Packet Size (Bytes)	Channel	Number of Rtx
Scenario 1: **Default Setting**	Burst (100, 10, 100)	28	26	6
Scenario 2: **Impact of Traffic Pattern**	Burst (100, 10, 100), Burst (200, 4, 500), Burst (100, 2, 1000), Synch (200, 1000)	28	26	6
Scenario 3: **Impact of Packet Size**	Burst (100, 10, 100)	28, 114	26	6
Scenario 4: **Impact of Channel**	Burst (100, 10, 100)	28	20, 26	6
Scenario 5: **Impact of Rtx count**	Burst (100, 10, 100)	28	26	0, 6

5.2 Performance Evaluation of Link Quality Estimators

The results obtained in this case study are related to perform evaluation of some LQEs. We conduct the evaluation using the DataAnlApp application which was useful to make the off-line analysis of data. This application was developed by our project-partner [23] and is part of RadiaLE framework. DataAnlApp application was developed in Matlab and it provides a GUI that connects to the database maintained by ExpCtrApp and processes this data, with two major functionalities: (i) generate a set of configurable and customized graphics that helps understanding the channel behavior; (ii) gives an assistance to RadiaLE users to evaluate the performance of their estimators.

Currently, DataAnlApp integrates a set of well-known LQEs, namely ETX, Four-bit, PRR, RNP, WMEWMA, and F-LQE. Other LQEs can be easily integrated and compared to existing LQEs, due to the flexibility and completeness of the collected empirical data.

We compared the performance of the LQEs in terms of two parameters, reliability and stability. Reliability refers to the ability of the LQE to correctly characterize the real link state. Stability refers to the ability to resist to transient (short-term) variations, also called fluctuations, in link quality.

5.2.1 Reliability Evaluation

The reliability of the LQEs under comparison can be evaluated by analyzing the distribution of their link quality estimates which can be, illustrated by a empirical Cumulative Distribution Function (CDF), and a scatter plot, and also their temporal behavior [7].

Figure 5.4 depicts the scatter plot showing that PRR, WMEWMA, and ETX, (these are all PRR-based LQEs), are either optimistic or overestimate the link quality. For instance, 80 % of the links in the network have a PRR and WMEWMA equals to 85 %, while 75 % of the links have an ETX equal to 1 [5].

On the other hand, figure 5.4 also shows that four-bit and RNP, which are based on RNP, are either pessimistic or underestimate the link quality. Almost 90 % of the links have RNP equal to 4 retransmissions (maximum value for RNP), which means that the link has very bad quality. Four-bit is less pessimistic than RNP as its computation accounts for PRR. This underestimation of RNP and four-bit is due to the fact that they are not able to determine if the packets are received after retransmissions or not. This discrepancy between PRR-based and RNP-based link quality estimates is justified by the fact that most of the packets transmitted over the link are correctly received (high PRR) but after a certain number of retransmissions (high RNP) [5].

Our proposed F-LQE provides reasonable link quality estimates (neither overestimates or underestimates the link quality). Furthermore, the distribution of link quality estimates is nearly an uniform distribution, which means that F-LQE is able to distinguish between links having different link qualities. These observations confirm the reliability of F-LQE. From figure 5.4 we can conclude the following: First, the higher β is, we have a more pessimistic F-LQE. Second and more important, by choosing β equal to 0.6, we get the distribution closest to the a uniform distribution, which justifies the choice of β equals to 0.6 in our study.

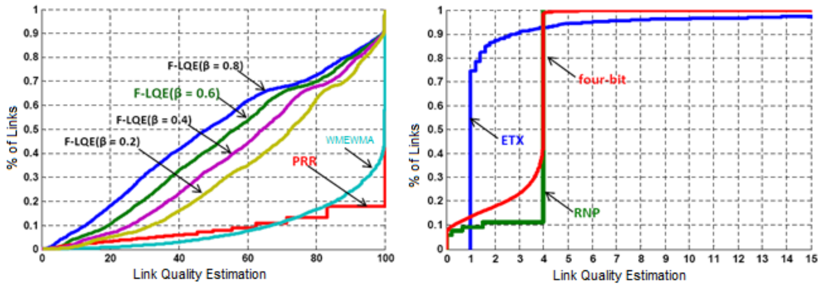


Figure 5.4: Empirical CDFs of LQEs (Default Setting)

From the scatter plot shown in figure 5.5, we can see that F-LQE estimates are more scattered than those of the other link estimators. For example, the RNP estimates are mostly aggregated to 4 retransmissions (the maximum). That means that two links assumed to have different qualities, may be aggregated to have almost the same qualities as when using RNP as LQE; and they would have different qualities when using F-LQE as LQE. The same thing holds for the rest of LQEs. This observation shows that F-LQE would surely perform better than the existing LQEs. Hence again, we show the reliability of F-LQE as it is able to provide a fine grain classification of links.

Figures 5.6, 5.7, 5.8, and 5.9 uses four different links to show the temporal behavior of each individual metric that constitutes F-LQE (ASL, SF, SPRR, and Asnr) and its overall behavior. It also presents the results from other existing LQEs. From this figure, it can be observed that all LQEs agree that the first link (figure 5.6) is of very good quality. This is expected since links of good quality are easy to estimate as they tend to be stable and symmetric. On the other hand, moderate and bad links, which are typically those of the transitional and the disconnected region respectively, are more difficult to characterize.

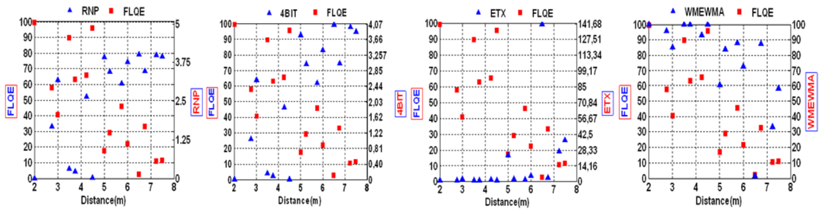


Figure 5.5: Scatter plot of each LQE according to distance (Default Setting)

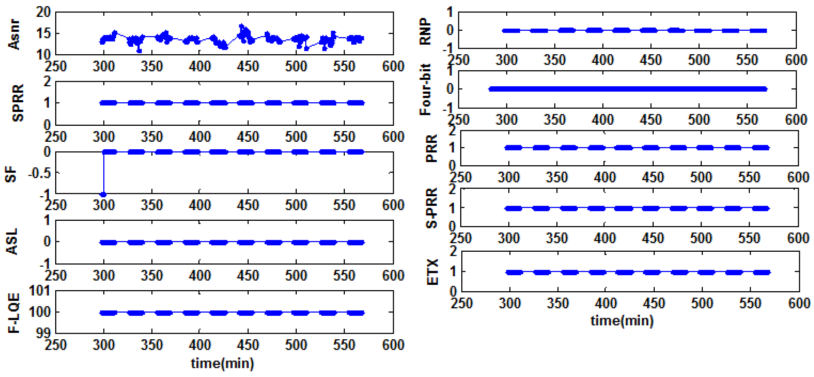


Figure 5.6: Temporal behavior of LQEs when faced with links with different qualities (Default Setting) for Very Good Link

Figure 5.7 shows how F-LQE outperforms the existing LQEs because they are not able to distinguish between links, especially good links and very good links. In fact, observing the temporal behavior of the link, until the time 3660 min (just before the link quality fluctuation). PRR, SPRR, and ETX are based on the PRR metric. They account for only one property: packet delivery. These LQEs based on PRR declare the link as of very good quality. The same link quality state is declared by RNP and four-bit, which are based on RNP and accounts for a unique link property. However, our link should not have a very good quality due to the low ASNR value. In fact, the measured

ASNR values are close to the receiver sensitivity. Consequently, the channel is of moderate quality, which prevents the link of being declared as "very good ". In addition, the good properties that the link has are likely due to the constructive interference effect. On the other hand, F-LQE detects the real link state by considering different link properties.

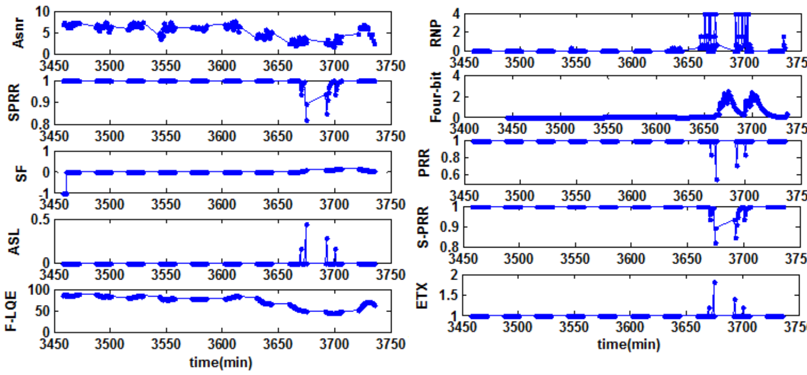


Figure 5.7: Temporal behavior of LQEs when faced with links with different qualities (Default Setting) for Good Link

From figure 5.8, we can observe how the LQEs based on PRR can overestimate link quality as they provide relatively high link quality estimates. The reason for this overestimation is the fact that LQEs based on PRR are only able to evaluate the packet delivery property and they are not aware of the number of retransmissions to deliver a packet. On the other hand the LQEs based on RNP, can underestimate link quality by providing low link quality estimates. This underestimation is due to the fact that each of these LQEs assesses the required packet retransmissions and are not able to determine if these packets are received after these retransmissions or not. More importantly, each of these LQEs assess a single and different link property. F-LQE estimates the link not as good as PRR-based estimators do, and not as bad as RNP-based estimators do. It takes into account different properties to provide a holistic characterization of the real link state.

Figure 5.9 is generally of bad quality. Furthermore, this link is a burst link, as its quality can turn to good (e.g. PRR equal to 1 and RNP equal to 0), yet in the short term. F-LQE is a stable LQE as it resists to these short-term link quality fluctuation whereas the other LQEs are not stable as their link quality estimates switch temporarily to very good estimates.

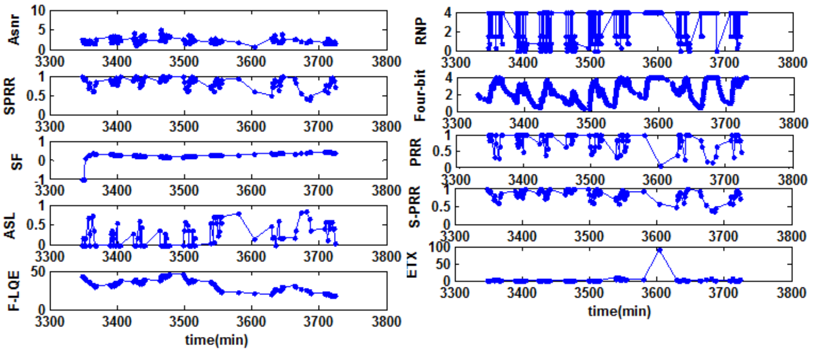


Figure 5.8: Temporal behavior of LQEs when faced with links with different qualities (Default Setting) for Moderate Link

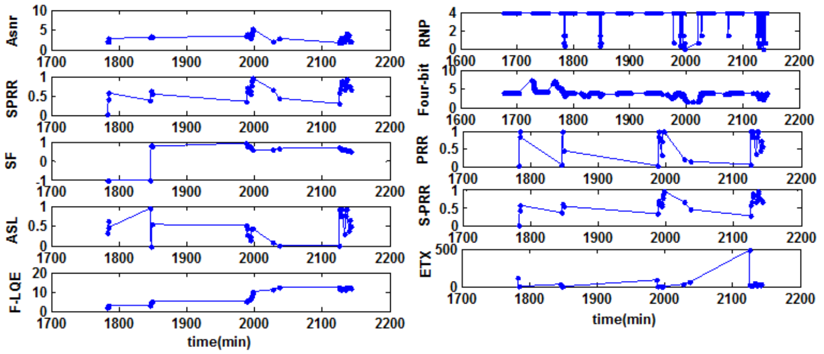
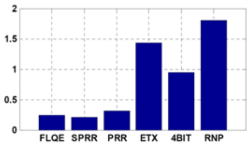


Figure 5.9: Temporal behavior of LQEs when faced with links with different qualities (Default Setting) for Moderate Link

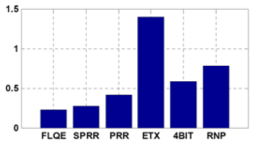
5.2.2 Stability Evaluation

A link may show transient link quality fluctuations due to factors mainly related to the environment, and also to the nature of low-power radios, which have been shown to be very prone to noise. LQEs should resist to these fluctuations and provide stable link quality estimates. This property is of paramount importance in WSNs. For instance, routing protocols do not have to reroute information when a link quality shows transient degradation, because rerouting is a very energy and time consuming operation.

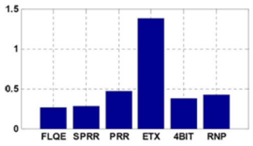
To reason about this issue, we measured the sensitivity of the LQEs to transient fluctuations by the coefficient of variation of its estimates. Figure 5.10 compares the sensitivity (stability) of F-LQE with that of PRR, ETX, SPRR, RNP, and four-bit, with respect to different settings (refer to table 5.1). According this figure, we retain two observations: First, generally, F-LQE is the most stable LQE. Second, except for ETX, PRR-based LQEs, i.e. PRR and SPRR, are more stable than RNP-based LQEs, i.e. RNP and four-bit. ETX is PRR-based, yet it is shown as unstable. The reason is that when the PRR tends to 0 (very bad link) the ETX will tend to infinity, which increases the standard deviation of ETX link estimates.



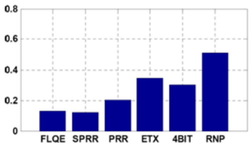
(**a**) Default settings



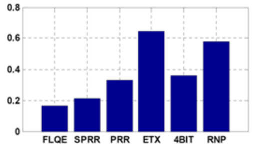
(**b**) Packet size : 114 Bytes



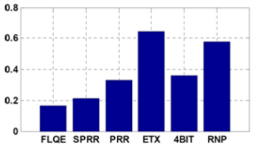
(**c**) Channel 20



(**d**) Burst(100,1000,2)



(**e**) Burst(200,500,4)



(**f**) Synch(200,1000)

Figure 5.10: Stability of LQEs, for different network settings

5.3 Concluding Remarks

Coming back to the RadiaLE testbed, it is important to highlight that it played a key role in the case study presented along this section. For instance, it was necessary to deploy and reprogram the nodes every day. Using our testbed, more specifically the *ExpCtrApp* application, it was much easier to select and to program the motes several times. This is very important for large-scale experiments, such as the experiments discussed here.

The testbed was also very important to help us selecting the network parameters for each performed experiment. Most important, the testbed was in charge to collect and log the packet-statistic from the sent and received packets during the experiments. All of the selected network parameters and the information about the experiments were stored in a database. This information was then made available to perform the off-line analyzes of data. The information about motes coordinates helped us to deploy the nodes all days in the same location, to draw the network topology and to analyze the link quality based on the distance among nodes. Finally, the testbed helps us to perform the off-line analyzes and compare the performance of the link quality estimators in WSN.

To summarize the results obtained in this case study it is possible to state that the existing estimators assess only a single link property, thus providing a partial view on the link quality. Further, sender side LQEs, namely RNP and four-bit are more responsive to link quality degradation than receiver side LQEs, i.e. PRR, WMEWMA, and ETX. To overcome this drawback it was proposed F-LQE, which combines four link quality properties namely, packet delivery, asymmetry, stability, and channel quality. Our conclusion was that F-LQE outperforms the existing LQEs.

Chapter 6

Conclusion

Experimental works give to researchers much more actuate results compared with simulation results. However, performing experiments in WSN without a proper tool (testbed) is very hard, especially considering tasks that are done manually when perform experiment without a testbed.

This work started with a detailed analysis and concluded that the testbeds proposed in the literature lack some expected features, such as logging the packet-statistics from sender and receiver side, choose a set of network parameters for each experiment, support only burst traffic with only one burst, support the nodes selection and programming, perform off-line analyzes, and be aware of motes locations.

The main contribution of this master thesis is proposing a new testbed, called RadiaLE that overcomes the mentioned deficiencies in existing testbeds. It automates the experimental evaluation of LQEs, as described in chapter 4. RadiaLE provides a friendly GUI enabling its users to configure, control, and perform the experiments. The proposed testbed has support to choose a set of parameters that can be tuned by the user. The proposed testbed was developed in collaboration with CISTER research group of ISEP.

Besides the creation of the testbed itself, this work also aimed to make experiments for performance evaluation of existing LQEs, without any interference from external factors, such as routing and collision. These experiments are presented in chapter 5. The collected data was useful to allow analyzing the behavior of existing LQEs in terms of two parameters, reliability and stability. Based on the obtained results we concluded that the existing LQEs assess only a single link property, thus providing a partial view on the link quality. Also the collected data was used to compare the behavior of a new LQE, called F-LQE that was developed in a related work. Based on results we conclude that F-LQE outperforms the existing LQEs.

For future work we plan the follow steps:

1. Extend the testbed capabilities, to allow perform experiments for analyzing the impact of LQEs in the routing layer
2. Also we plan extend the testbed to perform experiments for the evaluation of MAC protocols.

The RadiaLE testbed is left available publicly as an open-source at [38], together with all supporting documentation (e.g. installation and user guides), and publications. This work resulted in 1 submitted paper to the journal IEEE Transactions on Industrial Informatics with the title RadiaLE: a Framework for Benchmarking Link Quality Estimators, and also in 4 related publications:

1. F-LQE: A Fuzzy Link Quality Estimator for Wireless Sensor Networks [7];
2. Demo Abstract: A TestBed for the evaluation of Link Quality Estimators in WSNs [22];
3. A TestBed for the evaluation of Link Quality Estimators in Wireless Sensor Networks [5];
4. Demo Abstract: RadiaLE: a framework for benchmarking link quality Estimators [6].

Appendix A

Installing the RadiaLE Testbed

A.1 Install the Java Application Under Linux

A.1.1 System Requirements

We tested ExpCtrApp on:

1. Ubuntu 8.10
2. TinyOS 2.0.2
3. MySQL server 5.0.67
4. Java 1.6.0_10

A.1.2 Installation Steps

1. Install TinyOS: installation details can be found in:
   http://docs.tinyos.net/index.php/Installing_TinyOS_2.0.2
2. Install MySQL server: download and installation details can be found in:

http://dev.mysql.com/doc/refman/5.1/en/installing.html

3. Download our db.sql database from:

   http://www.das.ufsc.br/~denis/RadiaLE/database

4. Create two databases called experiment and backup. Follows the steps to create the databases:

   a. Logging

   ```
   $mysql -u root -p
   $insert the password
   ```

   b. Create the database

   ```
   $create database experiment
   $create database backup
   ```

   c. Select the database experiment

   ```
   $use experiment;
   ```

   d. Import the db.sql file to create the tables

   ```
   $source <path>/db.sql;
   ```

   e. Select the database backup

   ```
   $use backup;
   ```

   f. Import the db.sql file to create the tables

   ```
   $source <path>/db.sql;
   ```

5. Download the MySQL diver from:

   http://www.das.ufsc.br/~denis/RadiaLE/driver and save on:

   /usr/lib/jvm/default-java/jre/lib/ext

6. Install jfreechart, that enable create graph. The tutorial to install is available on:

   http://www.jfree.org/jfreechart/download/jfreechart-1.0.0-install.pdf

7. Download the Java application from:

   http://www.das.ufsc.br/~denis/RadiaLE/testbed

8. Open a prompt, go to the folder of the java application and type:

   $java run

9. Enjoy :)

A.2 Install the Java Application Under Windows

A.2.1 System Requirements

We tested ExpCtrApp on:

1. Windows XP
2. TinyOS 2.0.2
3. MySQL server 5.0.81-community-nt
4. Java 1.5.0_13

A.2.2 Installation Steps

1. Do steps 1, 2 and 3 of Linux installation
2. Open Mysql shell window
3. Do step 4 of Linux installation
4. Download the MySQL diver on

   http://www.das.ufsc.br/~denis/RadiaLE/driver and save on:

   C:\Program Files\Java\jdk1.5.0_13\jre\lib\ext
5. Do steps 6 and 7 of Linux installation
6. Put ExpCtrApp application under cygwin installation folder (for example you can put it under /opt folder)
7. The bash.exe need is saved in a path: c:\tinyos\cygwin\bin, otherwise find the bash.exe and change the path in the following java files:

a. Setting.java, line 542:

PROGRAMCMDW1 = ”c:\\tinyos\\cygwin\\bin\\bash.exe copyfile” + st1;

b. install.java, line 17:

PROGRAMCMDW1 = ”c:\\tinyos\\cygwin\\bin\\bash.exe programNodes” + id + ” ” + comport;

c. Open cygwin shell window and compile the java files, using the following command: $ javac setting.java

$ javac install.java

8. Open cygwin shell window
9. Go to the folder of ExpCtrApp application
10. type

    $ javac *.java

    $ java run
11. Enjoy :)

Appendix B

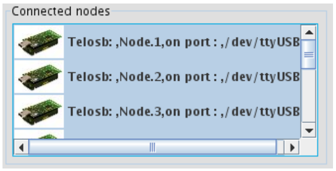
Using ExpCtrApp to Perform an Experiment

B.1 Steps to Perform an Experiment

The video in the follow link illustrates how to perform the experiment using our java tool: `www.das.ufsc.br/~denis/RadiaLE/video`

Follows a summary of the main steps to perform the experiment (illustrations are part of Figure B.7:

1. Connect a set of motes to the PC. They will be automatically detected by the application and displayed in the List of Connected Motes (picture below)



Telosb: ,Node.1,on port : ,/dev/ttyUSB
Telosb: ,Node.2,on port : ,/dev/ttyUSB
Telosb: ,Node.3,on port : ,/dev/ttyUSB

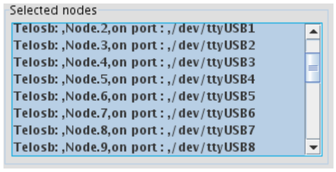
Figure B.1: List of Connected Nodes

2. Select the motes from the List of Connected Nodes, that will be involved in the experiment. To select the motes, the user need select the motes from the List of Connected Nodes just clicking with the left button of the mouse.

3. If you have connected/disconnected some motes to/from the PC, it is possible to click on the button to refresh the List of Connected Nodes.

4. Click on the button to effectively add the selected motes to the List of the Selected Nodes.

5. The List of the Selected Nodes (that will be involved in the experiment) appears in the following interface:



Selected nodes

```
Telosb: ,Node.2,on port : ,/dev/ttyUSB1
Telosb: ,Node.3,on port : ,/dev/ttyUSB2
Telosb: ,Node.4,on port : ,/dev/ttyUSB3
Telosb: ,Node.5,on port : ,/dev/ttyUSB4
Telosb: ,Node.6,on port : ,/dev/ttyUSB5
Telosb: ,Node.7,on port : ,/dev/ttyUSB6
Telosb: ,Node.8,on port : ,/dev/ttyUSB7
Telosb: ,Node.9,on port : ,/dev/ttyUSB8
```

Figure B.2: List of Selected Nodes

6. If you need to remove one or more motes from the List of Selected Nodes, then select the mote(s) from the List of the Selected Nodes and click in the button

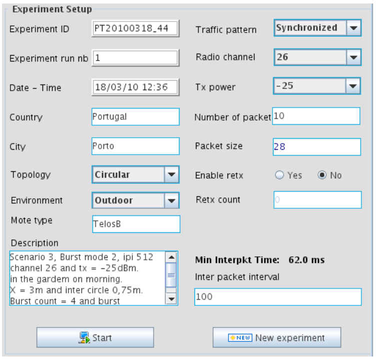
7. To install the nesC code on the motes:

   a. Click on the button Browse... to browse the source code to install on the motes:

   ExpCtrApp\Java_Code\build\telosb\main

   b. Click on the button Install to install the program in the List of Selected Nodes and wait while the source code is installed on the motes.

8. Set up the network and the experiment parameters using the following interface:



Experiment Setup

Experiment ID	PT20100318_44	Traffic pattern	Synchronized
Experiment run nb	1	Radio channel	26
Date - Time	18/03/10 12:36	Tx power	-25
Country	Portugal	Number of packet	10
City	Porto	Packet size	28
Topology	Circular	Enable retx	Yes [✓ No]
Environment	Outdoor	Retx count	0
Mote type	TelosB		

Description

Scenario 3, Burst mode 2, ipi 512
channel 26 and tx = -25dBm.
in the gardem on morning.
X = 3m and inter circle 0,75m.
Burst count = 4 and burst

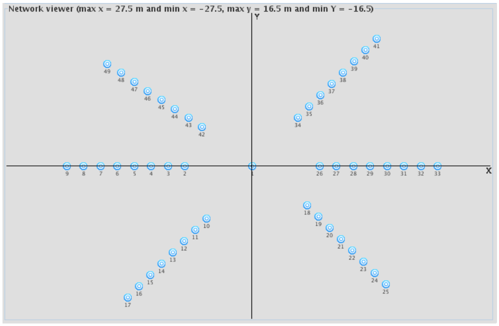
Min Interpkt Time: 62.0 ms

Inter packet interval: 100

Figure B.3: Setup the Network Parameters

9. Click on the button Start to start the experiment with the selected parameters.

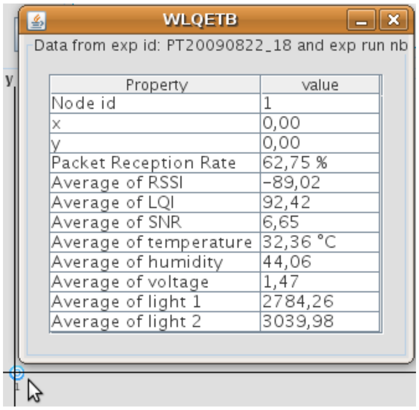
10. The network viewer shows the topology of the selected motes during the experiment.



Network viewer (max x = 27.5 m and min x = -27.5, max y = 16.5 m and min Y = -16.5)

Figure B.4: Network Viewer

11. If click on the icon, relevant information about the mote will be presented, as shown in the figure bellow:

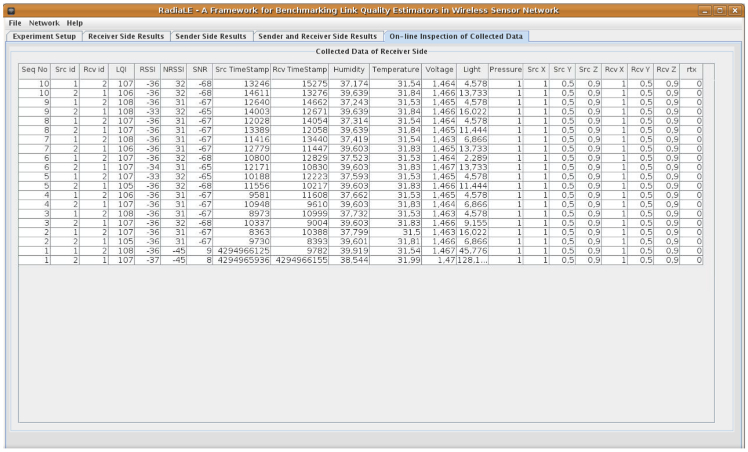


Data from exp id: PT20090822_18 and exp run nb

Property	value
Node id	1
x	0,00
y	0,00
Packet Reception Rate	62,75 %
Average of RSSI	-89,02
Average of LQI	92,42
Average of SNR	6,65
Average of temperature	32,36 °C
Average of humidity	44,06
Average of voltage	1,47
Average of light 1	2784,26
Average of light 2	3039,98

Figure B.5: Relevant Informations About the Mote

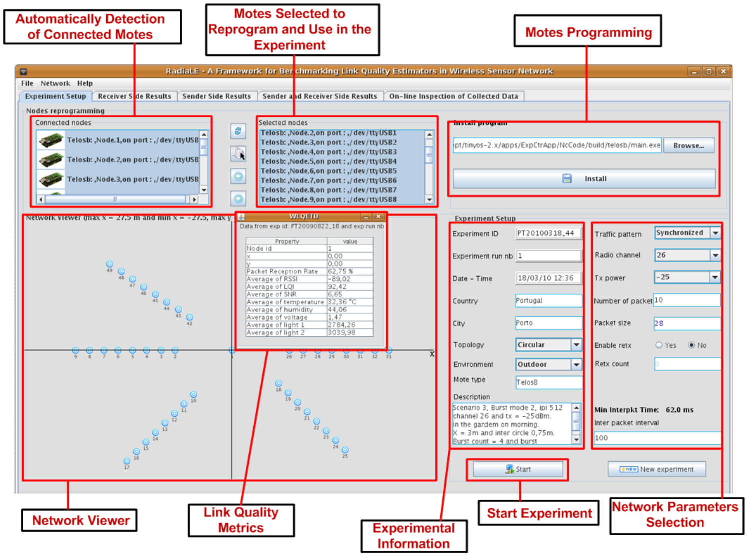
12. The Quick Database Inspect shows the collected data in real-time, as presented in the figure bellow:


RadiaLE - A Framework for Benchmarking Link Quality Estimators in Wireless Sensor Network
File
Network
Help
Experiment Setup
Receiver Side Results
Sender Side Results
Sender and Receiver Side Results
On-line Inspection of Collected Data

Collected Data of Receiver Side

Seq No	Src id	Rcv id	LQI	RSSI	NRSSI	SNR	Src TimeStamp	Rcv TimeStamp	Humidity	Temperature	Voltage	Light	Pressure	Src X	Src Y	Src Z	Rcv X	Rcv Y	Rcv Z	rtx
10	1	2	107	-36	32	-68	13246	15275	37,174	31,54	1,464	4,578	1	1	0,5	0,9	1	0,5	0,9	0
10	2	1	106	-36	32	-68	14611	13276	39,639	31,84	1,466	13,733	1	1	0,5	0,9	1	0,5	0,9	0
9	1	2	108	-36	31	-67	12640	14662	37,243	31,53	1,465	4,578	1	1	0,5	0,9	1	0,5	0,9	0
9	2	1	108	-33	32	-65	14003	12671	39,639	31,84	1,466	16,022	1	1	0,5	0,9	1	0,5	0,9	0
8	1	2	107	-36	31	-67	12028	14054	37,314	31,54	1,464	4,578	1	1	0,5	0,9	1	0,5	0,9	0
8	2	1	107	-36	31	-67	13389	12058	39,639	31,84	1,465	11,444	1	1	0,5	0,9	1	0,5	0,9	0
7	1	2	108	-36	31	-67	11416	13440	37,419	31,54	1,463	6,866	1	1	0,5	0,9	1	0,5	0,9	0
7	2	1	106	-36	31	-67	12779	11447	39,603	31,83	1,465	13,733	1	1	0,5	0,9	1	0,5	0,9	0
6	1	2	107	-36	32	-68	10800	12829	37,523	31,53	1,464	2,289	1	1	0,5	0,9	1	0,5	0,9	0
6	2	1	107	-34	31	-65	12171	10830	39,603	31,83	1,467	13,733	1	1	0,5	0,9	1	0,5	0,9	0
5	1	2	107	-33	32	-65	10188	12223	37,593	31,53	1,465	4,578	1	1	0,5	0,9	1	0,5	0,9	0
5	2	1	105	-36	32	-68	11556	10217	39,603	31,83	1,466	11,444	1	1	0,5	0,9	1	0,5	0,9	0
4	1	2	106	-36	31	-67	9581	11608	37,662	31,53	1,465	4,578	1	1	0,5	0,9	1	0,5	0,9	0
4	2	1	107	-36	31	-67	10948	9610	39,603	31,83	1,464	6,866	1	1	0,5	0,9	1	0,5	0,9	0
3	1	2	108	-36	31	-67	8973	10999	37,732	31,53	1,463	4,578	1	1	0,5	0,9	1	0,5	0,9	0
3	2	1	107	-36	32	-68	10337	9004	39,603	31,83	1,466	9,155	1	1	0,5	0,9	1	0,5	0,9	0
2	1	2	107	-36	31	-67	8363	10388	37,799	31,5	1,463	16,022	1	1	0,5	0,9	1	0,5	0,9	0
2	2	1	105	-36	31	-67	9730	8393	39,601	31,81	1,466	6,866	1	1	0,5	0,9	1	0,5	0,9	0
1	1	2	108	-36	-45	9	4294966125	9782	39,919	31,54	1,467	45,776	1	1	0,5	0,9	1	0,5	0,9	0
1	2	1	107	-37	-45	8	4294965936	4294966155	38,544	31,99	1,47	128,1...	1	1	0,5	0,9	1	0,5	0,9	0

Figure B.6: GUI to Visualize the Data in Real-Time


Automatically Detection of Connected Motes
Motes Selected to Reprogram and Use in the Experiment
Motes Programming
RadiaLE - A Framework for Benchmarking Link Quality Estimators in Wireless Sensor Network
File
Network
Help
Experiment Setup
Receiver Side Results
Sender Side Results
Sender and Receiver Side Results
On-line Inspection of Collected Data
Browse...
Install
Start
New experiment
Network Viewer
Link Quality Metrics
Experimental Information
Start Experiment
Network Parameters Selection

Nodes reprogramming

Connected nodes

Telosb: ,Node.1,on port : ,/dev/ttyUSB
Telosb: ,Node.2,on port : ,/dev/ttyUSB
Telosb: ,Node.3,on port : ,/dev/ttyUSB

Selected nodes

Telosb: ,Node.2,on port : ,/dev/ttyUSB1
Telosb: ,Node.3,on port : ,/dev/ttyUSB2
Telosb: ,Node.4,on port : ,/dev/ttyUSB3
Telosb: ,Node.5,on port : ,/dev/ttyUSB4
Telosb: ,Node.6,on port : ,/dev/ttyUSB5
Telosb: ,Node.7,on port : ,/dev/ttyUSB6
Telosb: ,Node.8,on port : ,/dev/ttyUSB7
Telosb: ,Node.9,on port : ,/dev/ttyUSB8

Install program

pt/tinyos-2.x/apps/ExpCtrApp/NcCode/build/telosb/main.exe

Network viewer (max x = 27.5 m and min x = -27.5, max y

WLQETB

Data from exp id: PT20090822_18 and exp run nb

Property	value
Node id	1
x	0,00
y	0,00
Packet Reception Rate	62,75 %
Average of RSSI	-89,02
Average of LQI	92,42
Average of SNR	6,65
Average of temperature	32,36 °C
Average of humidity	44,06
Average of voltage	1,47
Average of light 1	2784,26
Average of light 2	3039,98

Experiment Setup

Experiment ID	PT20100318_44
Experiment run nb	1
Date - Time	18/03/10 12:36
Country	Portugal
City	Porto
Topology	Circular
Environment	Outdoor
Mote type	TelosB

Description

Scenario 3, Burst mode 2, ipi 512
channel 26 and tx = -25dBm.
in the gardem on morning.
X = 3m and inter circle 0,75m.
Burst count = 4 and burst

Traffic pattern	Synchronized
Radio channel	26
Tx power	-25
Number of packet	10
Packet size	28
Enable retx	Yes [✓ No]
Retx count	0

Min Interpkt Time: 62.0 ms

Inter packet interval

100

Figure B.7: GUI to Perform Experiments

Appendix C

Using ExpCtrApp to Analyze the Results

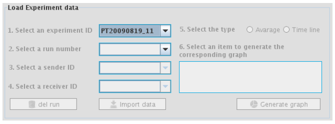
C.1 Steps to perform the analyze of results

As the experiment finishes, you can use the following functionalities to make an off-line and quick analysis of the experiment data. Experiment data analysis using the ExpCtrApp can be used just to give an idea about the data collected during the experiment. A thorough analysis is provided by the DataAnlApp Matlab application.

Figure C.6 depicts the interface used to perform the off-line analyze. Follows a summary of the main steps to analyze the results using the ExpCtrApp java application.

1. ExpCtrApp provides the following interfaces for the experiment data analysis: Receiver Results, Sender Results, and Sender and Receiver Results. Select for instance the interface Receiver Results interface. This interface looks as shown in the incoming picture.

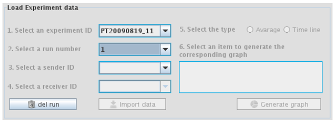
2. Select an Experiment Id, as show the figure bellow:


Load Experiment data

1. Select an experiment ID: PT20090819_11
2. Select a run number:
3. Select a sender ID:
4. Select a receiver ID:
5. Select the type: Avarage / Time line
6. Select an item to generate the corresponding graph

Figure C.1: Select Experiment Identifier

3. Select a Run Number, as depict the follow figure. Note that a single experiment (defined by a set of settings) can be executed many times.

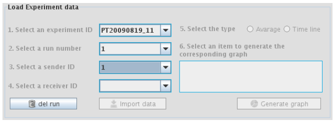

Load Experiment data

1. Select an experiment ID: PT20090819_11
2. Select a run number: 1
3. Select a sender ID:
4. Select a receiver ID:
5. Select the type: Avarage / Time line
6. Select an item to generate the corresponding graph

Figure C.2: Select Experiment Run Number

4. To delete data related to a given experiment, just select the run number and click on the button del run

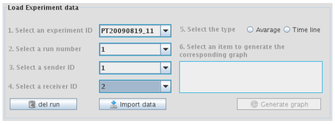
5. Select a Sender Id, one specific or all senders, as illustrated in the follow figure


Load Experiment data

1. Select an experiment ID: PT20090819_11
2. Select a run number: 1
3. Select a sender ID: 1
4. Select a receiver ID:
5. Select the type: Avarage / Time line
6. Select an item to generate the corresponding graph

Figure C.3: Select Sender Identifier

6. Select a Receiver id, one specific of all receiver, as shows the follow figure


Load Experiment data

1. Select an experiment ID: PT20090819_11
2. Select a run number: 1
3. Select a sender ID: 1
4. Select a receiver ID: 2
5. Select the type: Avarage / Time line
6. Select an item to generate the corresponding graph

Figure C.4: Select Receiver Identifier

7. The user can import the data according the selected information. Just clicking the button Import data, that import the data to visualization on the interface.

8. To create some graphs.

a. Select a type of graph 5. Select the type [✓ Avarage] Time line

b. Select the metric

PACKET RECEPTION RATE
PACKET RECEPTION RATE - BAR
REGION OF MOTES
LQI

c. Click on the button Generate graph to create the graph.

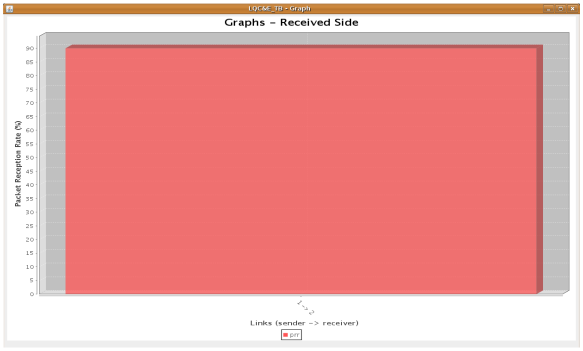
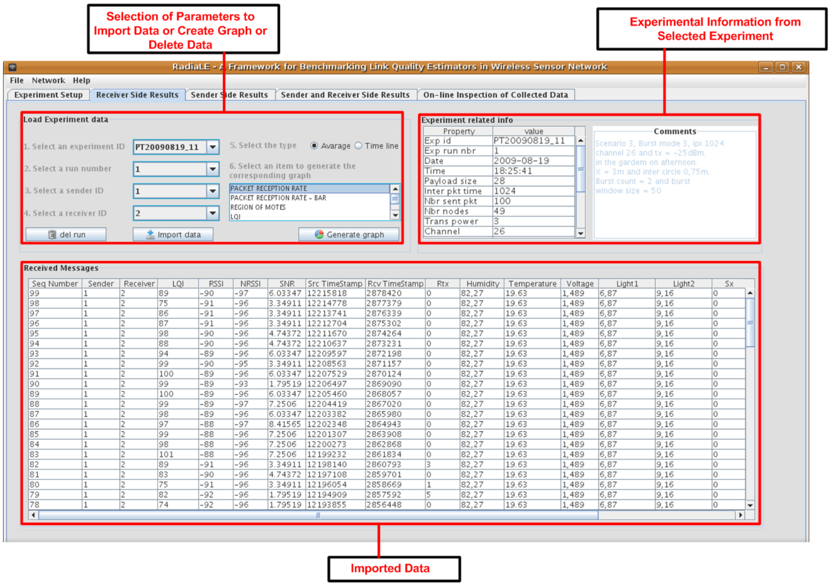


Figure C.5: Graph of PRR of link 1 to 2


Selection of Parameters to Import Data or Create Graph or Delete Data
Experimental Information from Selected Experiment
Imported Data
RadiaLE - A Framework for Benchmarking Link Quality Estimators in Wireless Sensor Network
File
Network
Help
Experiment Setup
Receiver Side Results
Sender Side Results
Sender and Receiver Side Results
On-line Inspection of Collected Data
del run
Import data
Generate graph

Load Experiment data

1. Select an experiment ID: PT20090819_11
2. Select a run number: 1
3. Select a sender ID: 1
4. Select a receiver ID: 2
5. Select the type: (●) Avarage ○ Time line
6. Select an item to generate the corresponding graph: PACKET RECEPTION RATE; PACKET RECEPTION RATE - BAR; REGION OF MOTES; LQI

Experiment related info

Property	value
Exp id	PT20090819_11
Exp run nbr	1
Date	2009-08-19
Time	18:25:41
Payload size	28
Inter pkt time	1024
Nbr sent pkt	100
Nbr nodes	49
Trans power	3
Channel	26

Comments

Scenario 3, Burst mode 3, ipi 1024
channel 26 and tx = -25dBm.
in the gardem on afternoon.
X = 3m and inter circle 0,75m.
Burst count = 2 and burst
window size = 50

Received Messages

Seq Number	Sender	Receiver	LQI	RSSI	NRSSI	SNR	Src TimeStamp	Rcv TimeStamp	Rtx	Humidity	Temperature	Voltage	Light1	Light2	Sx
99	1	2	89	-90	-97	6.03347	12215818	2878420	0	82,27	19.63	1,489	6,87	9,16	0
98	1	2	75	-91	-96	3.34911	12214778	2877379	0	82,27	19.63	1,489	6,87	9,16	0
97	1	2	86	-91	-96	3.34911	12213741	2876339	0	82,27	19.63	1,489	6,87	9,16	0
96	1	2	87	-91	-96	3.34911	12212704	2875302	0	82,27	19.63	1,489	6,87	9,16	0
95	1	2	98	-90	-96	4.74372	12211670	2874264	0	82,27	19.63	1,489	6,87	9,16	0
94	1	2	88	-90	-96	4.74372	12210637	2873231	0	82,27	19.63	1,489	6,87	9,16	0
93	1	2	94	-89	-96	6.03347	12209597	2872198	0	82,27	19.63	1,489	6,87	9,16	0
92	1	2	99	-90	-95	3.34911	12208563	2871157	0	82,27	19.63	1,489	6,87	9,16	0
91	1	2	100	-89	-96	6.03347	12207529	2870124	0	82,27	19.63	1,489	6,87	9,16	0
90	1	2	99	-89	-93	1.79519	12206497	2869090	0	82,27	19.63	1,489	6,87	9,16	0
89	1	2	100	-89	-96	6.03347	12205460	2868057	0	82,27	19.63	1,489	6,87	9,16	0
88	1	2	99	-89	-97	7.2506	12204419	2867020	0	82,27	19.63	1,489	6,87	9,16	0
87	1	2	98	-89	-96	6.03347	12203382	2865980	0	82,27	19.63	1,489	6,87	9,16	0
86	1	2	97	-88	-97	8.41565	12202348	2864943	0	82,27	19.63	1,489	6,87	9,16	0
85	1	2	99	-88	-96	7.2506	12201307	2863908	0	82,27	19.63	1,489	6,87	9,16	0
84	1	2	98	-88	-96	7.2506	12200273	2862868	0	82,27	19.63	1,489	6,87	9,16	0
83	1	2	101	-88	-96	7.2506	12199232	2861834	0	82,27	19.63	1,489	6,87	9,16	0
82	1	2	89	-91	-96	3.34911	12198140	2860793	3	82,27	19.63	1,489	6,87	9,16	0
81	1	2	83	-90	-96	4.74372	12197108	2859701	0	82,27	19.63	1,489	6,87	9,16	0
80	1	2	75	-91	-96	3.34911	12196054	2858669	1	82,27	19.63	1,489	6,87	9,16	0
79	1	2	82	-92	-96	1.79519	12194909	2857592	5	82,27	19.63	1,489	6,87	9,16	0
78	1	2	74	-92	-96	1.79519	12193855	2856448	0	82,27	19.63	1,489	6,87	9,16	0

Figure C.6: GUI to Perform Off-line Analysis of Data

Bibliography

[1] 802.15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs). *IEEE standard for Information Technology*, 2006.

[2] Zigbee Alliance. Zigbee specification, February 2010. Available at: `http://www.zigbee.org/` and access at 22 feb. 2010.

[3] Atmel. Atmega1281 8-bit microcontroller datasheet, March 2010. Available at: `http://www.atmel.com` and access at 01 mar. 2010.

[4] N. Baccour, A. Koubâa, M. B. Jamâa, H. Youssef, M. Zuniga, and M. Alves. A Comparative Simulation Study of Link Quality Estimators in Wireless Sensor Networks. In *Proceedings of the 17th IEEE/ACM International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS 09)*. South Kensington Campus, Imperial College London, September 2009.

[5] N. Baccour, M. B. Jamâa, D. Rosário A. Koubâa, , H. Youssef, M. Alves, and L. B. Becker. A TestBed for the evaluation of Link Quality Estimators in Wireless Sensor Networks. In *Proceedings of the ACS/IEEE Workshop Future Trends on Ad-hoc and Sensor Networks (FT-ASN 2010)*. Hammamet, Tunisia, May 2010.

[6] N. Baccour, M. B. Jamâa, D. Rosário, A. Koubâa, M. Alves, L. B. Becker, H. Youssef, and H. Fotouhi. Demo Abstract: RadiaLE: a framework for benchmarking link quality estimators. In *Proceedings of the*

7th European Conference on Wireless Sensor Networks (EWSN 2010). University of Coimbra, Coimbra, Portugal, February 2010.

[7] N. Baccour, A. Koubâa, H. Youssef, M. B. Jamâa, D. Rosário, L. B. Becker, and M. Alves. F-LQE: A Fuzzy Link Quality Estimator for Wireless Sensor Networks. In *Proceedings of the 7th European Conference on Wireless Sensor Networks (EWSN 2010)*, pages 240–255. University of Coimbra, Coimbra, Portugal, February 2010.

[8] M. Ceriotti, L. Mottola, G.P. Picco, A.L. Murphy, S. Guna, M. Corra, M. Pozzi, D. Zonta, and P. Zanon. Monitoring heritage buildings with wireless sensor networks: The Torre Aquila deployment. In *Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN) - SPOTS track*, pages 277–288. IEEE Computer Society, 2009.

[9] A. Cerpa, N. Busek, and D. Estrin. SCALE: A tool for simple connectivity assessment in lossy environments. Technical report, CENS, UCLA, Sep 2003.

[10] A. Cerpa, J. L. Wong, M. Potkonjak, and D. Estrin. Temporal properties of low power wireless links: modeling and implications on multi-hop routing. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc 05)*, pages 414–425. New York, Ny, USA: ACM, 2005.

[11] Chipcom. Cc2420 datasheet, March 2010. Available at: `http://focus.ti.com/lit/ds/symlink/cc2420.pdf` and access at 01 mar. 2010.

[12] B. N. Chun, P. Buonadonna, A. AuYoung, C. Ng, D. C. Parkes, J. Shneidman, A. C. Snoeren, and A. Vahdat. Mirage: A microeconomic resource allocation system for sensornet testbeds. In *Proceedings of the 2nd IEEE workshop on Embedded Networked Sensors*, pages 19–28, may 2005.

[13] D. S. J. D. Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of*

the 9th annual international conference on Mobile computing and networking (MobiCom 03), pages 134–146. New York, NY, USA: ACM, 2003.

[14] J. Elson, S. Bien, N. Busek, V. Bychkovskiy, A. Cerpa, D. Ganesan, L. Girod, B. Greenstein, T. Schoellhammer, T. Stathopoulos, and D. Estrin. Emstar: An environment for developing wireless embedded systems software. Technical report, Center for Embedded Networked Sensing (CENS), 2003.

[15] A. Eswaran, A. Rowe, and R. Rajkumar. Nano-rk: an energy-aware resource-centric rtos for sensor networks. In *Real-Time Systems Symposium, 2005. RTSS 2005. 26th IEEE International*, pages 10 pp. –265, dec. 2005.

[16] R. Fonseca, O. Gnawali, K. Jamieson, and P. Levis. Four-bit wireless link estimation. In *Proceedings of the Sixth Workshop on Hot Topics in Networks (HotNets VI)*, 2007.

[17] Communication Architecture for Real-Time Mobile Cooperative Applications. Acervo web site, October 2009. Available at: `http://acervo.das.ufsc.br/` and access at 10 oct. 1009.

[18] D. Gay, P. Levis, R. Von Behren, M. Welsh, E. Brewer, and D. Culler. The nesC language: A holistic approach to networked embedded systems. In *Proceedings of the ACM SIGPLAN 2003 conference on Programming language design and implementation (PLDI 03)*, pages 1–11. New York, NY, USA: ACM, 2003.

[19] V. Handziski, A. Köpke, A. Willig, and A. Wolisz. TWIST: a scalable and reconfigurable testbed for wireless indoor experiments with sensor networks. In *Proceedings of the 2nd international workshop on Multi-hop ad hoc networks: from theory to reality (REALMAN 06)*, pages 63–70. New York, NY, USA: ACM, 2006.

[20] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. In *Proceedings of the*

Ninth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-IX), pages 93–104. Cambridge, MA, USA, 2000.

[21] Texas Instruments. Msp430 microcontroller datasheet, March 2010. Available at: `http://focus.ti.com/lit/ds/symlink/msp430f149.pdf` and access at 01 mar. 2010.

[22] M. B. Jamâa, N. Baccour, A. Koubâa, D. Rosário, M. Alves, L. B. Becker, H. Youssef, and M. Jmaiel. Demo Abstract: A TestBed for the evaluation of Link Quality Estimators in WSNs. In *Proceedings of the First International School on Cyber-Physical and Sensor Networks (SensorNets 09)*. CES/ENIS, Monastir, Tunisia, December 2009.

[23] Maissa Ben Jamâa. Benchmarking link quality estimators in wsn. Master's thesis, Ecole Nationale des Ingénieurs de Sfax, Sfax, Tunisia, 2010.

[24] H. Karl and A. Willig. *Protocols and architectures for wireless sensor networks*. John Wiley & Sons Inc, 2005.

[25] A. Koubâa, R. Severino, M. Alves, and E. Tovar. Improving Quality-of-Service in Wireless Sensor Networks by mitigating hidden-node collisions. *IEEE Transactions on Industrial Informatics, Special Issue on Real-Time and Embedded Networked Systems*, 5(3), aug 2009.

[26] P. Levis, N. Lee, M. Welsh, and D. Culler. TOSSIM: Accurate and scalable simulation of entire TinyOS applications. In *Proceedings of the 1st international conference on Embedded networked sensor systems (SenSys 03)*, pages 126–137. New York, NY, USA: ACM, 2003.

[27] P. Santi. Topology control in wireless ad hoc and sensor networks. *ACM Computing Surveys (CSUR)*, 37(2):164–194, 2005.

[28] Victor Shnayder, Bor-rong Chen, Konrad Lorincz, Thaddeus R. F. Fulford Jones, and Matt Welsh. Sensor networks for medical care. In *Proceedings of the 3rd international conference on Embedded networked sensor systems (SenSys 05)*, pages 314–314, New York, NY, USA, 2005.

[29] M. M. Sobral and L. B. Becker. Communication Architecture for Real-Time Mobile Cooperative Applications. In *Proceedings of the International Workshop on Dependable Network Computing and Mobile Systems*. Naples, 2008.

[30] M. M. Sobral and L. B. Becker. A wireless hybrid contention/TDMA-based MAC for real-time mobile application. In *Proceedings of the ACM Symposium on Applied Computing (SAC 08)*, pages 284–288, New York, NY, USA, 2008.

[31] M. M. Sobral and L. B. Becker. A Real-Time Subject Routing Protocol for MANETs. In *Proceedings of the 7th International Workshop on Real-Time Networks (RTN 08)*. Prague, Czech Republic, 2008.

[32] K. Srinivasan, M. A. Kazandjieva, M. Jain, E. Kim, and P. Levis. Demo Abstract: SWAT: enabling wireless network measurements. In *Proceedings of the 6th ACM conference on Embedded network sensor systems (SenSys 08)*, pages 395–396. New York, NY, USA: ACM, 2008.

[33] OPNET Technologies, October 2009. Available at: `http://www.opnet.com/` and access at 10 oct. 2009.

[34] Crossbow Technology. Mib510 datasheet, March 2010. Available at: `http://www.xbow.com/products/product_pdf_files/wireless_pdf/mib510ca_datasheet.pdf` and access at 01 mar. 2010.

[35] Crossbow Technology. Mib520 datasheet, March 2010. Available at: `http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MIB520_Datasheet.pdf` and access at 01 mar. 2010.

[36] Crossbow Technology. Micaz datasheet, March 2010. Available at: `http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAZ_Datasheet.pdf` and access at 01 mar. 2010.

[37] Crossbow Technology. Telosb datasheet, March 2010. Available at: `http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/TelosB_Datasheet.pdf` and access at 01 mar. 2010.

[38] RadiaLE Benchmarking Tool, February 2010. Available at: `http://www.open-LQE.net` and access at 22 feb. 2010.

[39] A. Varga. The OMNeT++ discrete event simulation system. In *Proceedings of the European Simulation Multiconference (ESM 01)*, pages 319–324, 2001.

[40] G. Werner-Allen, J. Johnson, M. Ruiz, J. Lees, and M. Welsh. Monitoring volcanic eruptions with a wireless sensor network. In *Proceeedings of the Second European Workshop on Wireless Sensor Networks (EWSN 2005)*, pages 108–120, jan 2005.

[41] G. Werner-Allen, P. Swieskowski, and M. Welsh. Motelab: A wireless sensor network testbed. In *Proceedings of the Fourth International Conference on Information Processing in Sensor Networks (IPSN 05), Special Track on Platform Tools and Design Methods for Network Embedded Sensors (SPOTS)*, 2005.

[42] A. Woo and D. Culler. Evaluation of Efficient Link Reliability Estimators for Low-Power Wireless Networks. Technical Report UCB/CSD-03-1270, EECS Department, University of California, Berkeley, 2003. Available at: `http://www.eecs.berkeley.edu/Pubs/TechRpts/2003/6239.html`.

[43] M. Zuniga and B. Krishnamachari. Analyzing the transitional region in low power wireless links. In *Proceedings of the First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (IEEE SECON)*, volume 2004, pages 517–526, 2004.