Pós-Graduação em Ciência da Computação

"AN ENERGY-AWARE MULTIPATH ROUTING EXTENSION FOR HETEROGENEOUS AD HOC NETWORKS"

Por

JOSIAS BARBOSA DE LIMA JUNIOR

Dissertação de Mestrado

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE, MAIO/2013

UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

JOSIAS BARBOSA DE LIMA JUNIOR

"AN ENERGY-AWARE MULTIPATH ROUTING EXTENSION FOR HETEROGENEOUS AD HOC NETWORKS"

ESTE TRABALHO FOI APRESENTADO À PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO DO CENTRO DE INFORMÁTICA DA UNIVERSIDADE FEDERAL DE PERNAMBUCO COMO REQUISITO PARCIAL PARA OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIA DA COMPUTAÇÃO.

ORIENTADOR(A): Judith Kelner

RECIFE, MAIO/2013

Catalogação na fonte
Bibliotecária Jane Souto Maior, CRB4-571

Lima Junior, Josias Barbosa de
An energy-aware multipath routing extension for heterogeneous ad hoc networks. / Josias Barbosa de Lima Junior. - Recife: O Autor, 2013.
x, 138 folhas: fig., tab.

Orientador: Judith Kelner.
Dissertação (mestrado) - Universidade Federal de Pernambuco. CIn, Ciência da Computação, 2013.

Inclui bibliografia, anexo e apêndice.

1. Redes de computadores. 2. Comunicação móvel sem fio. 3. Roteamento. I. Kelner, Judith (orientadora). II. Título.

004.6 CDD (23. ed.) MEI2013 – 071

Dissertação de Mestrado apresentada por **Josias Barbosa de Lima Junior** à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título "**Uma Extensão Baseada em Múltiplos Caminhos para Redes Heterogêneas ad hoc com Percepção Energética"** orientada pela Profa. Judith Kelner e aprovada pela Banca Examinadora formada pelos professores:

Prof. Stênio Flávio de Lacerda Fernandes
Centro de Informática / UFPE

Prof. Eduardo Luzeiro Feitosa
Instituto de Computação / UFAM

Profa. Judith Kelner
Centro de Informática / UFPE

Visto e permitida a impressão.
Recife, 23 de abril de 2013.

Profa. Edna Natividade da Silva Barros
Coordenadora da Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.

ACKNOWLEGMENTS

First and foremost, I offer my sincerest gratitude and love to my parents, Josias Barbosa and Maria Valda, who have supported me during all those years with patience, support and love. To my sister, Michelle, who stood with me at all times, always giving me support. To the rest of my family, my cousins, uncles and aunts, thank you. I cannot forget to sincerely thank my advisor Prof. Judith Kelner and Prof. Djamel Sadok, for all patience and support within this dissertation and at GPRT, providing a great opportunity for my personal and professional growth. To all the SEFIN team that helped me in many difficult situations and granted me the chance to successfully complete this work. To all those who have been with me on this journey, all friends that somehow supported me and helped on this personal realization.

I would like to thank God. Furthermore, I would like to thank the SPORT Club do Recife. Finally, I am particularly grateful to my soul mate, Ingrid Nogueira Gomes, for her help, for her love and for assist me in my life. Without her help, this dissertation would never have been possible. I love you so much. Thanks!!

Sincerely, Josias Junior.

Abstract

Recent years have witnessed the emergence of new communication techniques in Computer Science that use both wireless technologies and self-organizing features. Their combination eliminates the need for using pre-defined wired structures and prior configurations. In this work, we develop a simulated version, using the network simulator 3 (ns-3), of the Heterogeneous Technologies Routing (HTR) framework that is suitable for interconnecting devices in a heterogeneous ad hoc network, extending its supported heterogeneous technologies with the addition of WiMAX and LTE devices, proposes an extension to enable multipath routing over this framework and investigates the impact of tuning routing parameters on convergence interval and energy consumption. Although a large number of works exist that investigate the tuning of routing parameter settings, to the best of our knowledge, none of them investigate the impact of these on protocol convergence and energy consumption. Multipath HTR routing, the extension we propose, offers several benefits such as load balancing, fault tolerance, routing loop prevention, energy-conservation, low end-to-end delay, and congestion avoidance, among others. This work performs a comparative analysis of the proposed HTR extension, with the baseline HTR, and the widely used Optimized Link State Routing (OLSR) protocol. Moreover, we investigate the impacts of tuning the HELLO refresh interval and perform a comparative analysis of the tuned HTR with the OLSR protocol. Both evaluations are validated through the simulation of heterogeneous technologies such as WiMAX, 3GPP LTE and Wi-Fi. Results show that the multipath extension effectively improves the data delivery ratio, and reduces the end-to-end delay without major impact on network energy consumption. For the tuned HTR, results show that varying the HELLO refresh interval can improve the convergence of the protocol and reduce the energy consumption.

Keywords: Wireless Mobile Communication; Mobile Ad hoc Networks (MANET); Heterogeneous technologies; multipath routing; ns-3; WiMAX; Wi-Fi; LTE; simulation; tuning; settings; soft-state; convergence;

RESUMO

Recentemente, novas técnicas de comunicação surgiram que usam tecnologia sem fio e são capazes de se autoconfigurar. A combinação desses fatores elimina a necessidade de utilizar estruturas cabeadas e configurações pré-definidas. Neste trabalho, o autor desenvolve uma versão simulada, através do simulador de rede “network Simulator 3” (ns-3), do arcabouço “Heterogeneous Technologies Routing” (HTR), que se propõe à interconectar dispositivos em redes heterogêneas ad hoc, estendendo o seu suporte às tecnologias heterogêneas com a adição de dispositivos WiMAX e LTE, propõe uma extensão para fornecer um roteamento baseado em múltiplos caminhos (“Multipath”) e investiga o impacto de modificar os parâmetros de configuração do roteamento no tempo de convergência da rede e consumo de energia. Apesar de um grande número de obras existentes que investigam o impacto da mudança de parâmetros de configuração do roteamento, no meu conhecimento, nenhum deles investiga o impacto destes no tempo de convergência do protocolo e consumo de energia. O “Multipath HTR”, a extensão proposta, oferece vários benefícios como balanceamento de carga, tolerância a falhas, prevenção de “loops” de roteamento, conservação de energia, baixo atraso fim-a-fim, e evita o congestionamento, entre outros. Este trabalho faz uma análise comparativa da extensão ao HTR proposta, com a base do HTR, e o protocolo amplamente utilizado “Optimized Link State Routing” (OLSR). Além disso, o esse trabalho investiga o impacto de variar o intervalo de envio de mensagens de HELLO e realiza uma análise comparativa do HTR modificado (“Tuned HTR”) com o protocolo OLSR. As duas avaliações são realizadas através de simulação usando tecnologias heterogêneas como WiMAX, 3GPP LTE e Wi-Fi. Resultados mostram que a extensão de múltiplos caminhos proposta melhora a taxa de transmissão de dados, e reduz o atraso fim-a-fim sem maiores impactos no consumo de energia da rede. Para o “Tuned HTR”, resultados mostram que a variação do intervalo de envio de mensagens de HELLO pode melhorar a convergência do protocolo e reduz o consumo de energia.

Palavras-chave: Comunicação móvel sem fio; “Mobile Ad Hoc Networks (MANET)”; Tecnologias heterogêneas; Roteamento por múltiplos caminhos; ns-3; WiMAX; LTE; simulação; “tunning”; configurações; “soft-state”; convergência;

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 INTRODUCTION

The popularity of mobile communication is constantly increasing [1]. The worldwide internet mobile traffic is expected to overtake the desktop internet traffic by 2014 [2], which means that more users will be accessing the Internet through their mobile devices (e.g. Smartphone and Tablets) than through their Personal Computers (PCs). In [3], the author shows that mobile devices now accounts for 12 percent of all Internet traffic (see Figure 1). Furthermore, this author reveals that PCs shipments continue to drop and also that Smartphones and Tablets shipments overtook those of PCs since 2011 (see Figure 2).

Figure 1: Worldwide Mobile Traffic as percentage of Total Internet Traffic

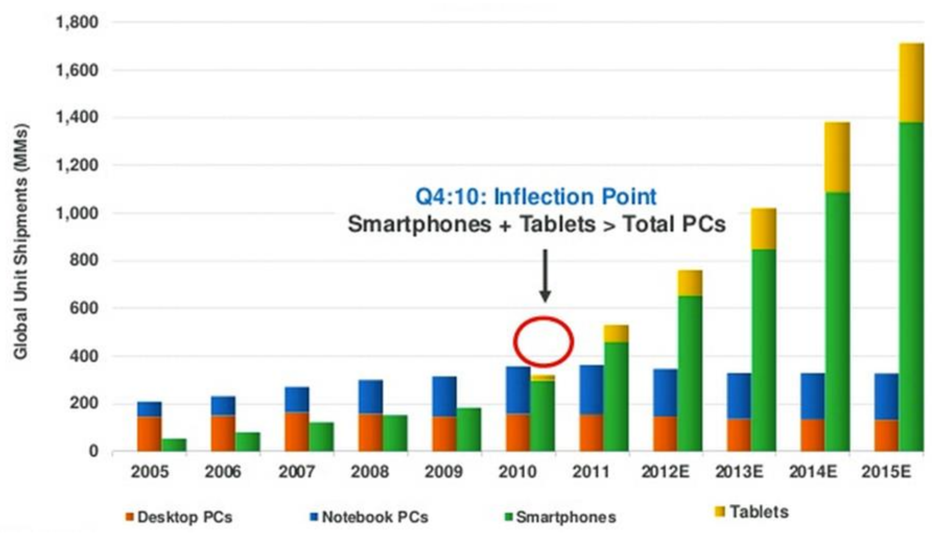


Figure 2: Global Unit Shipments of Desktop PCS + Notebook PCS vs. Smartphones + Tablets

Mobile devices are rapidly becoming the preferred computing platform for many (see Figure 3) and this phenomenon has already been experienced in some countries, such as China [4] and India [5] (see Figure 4).

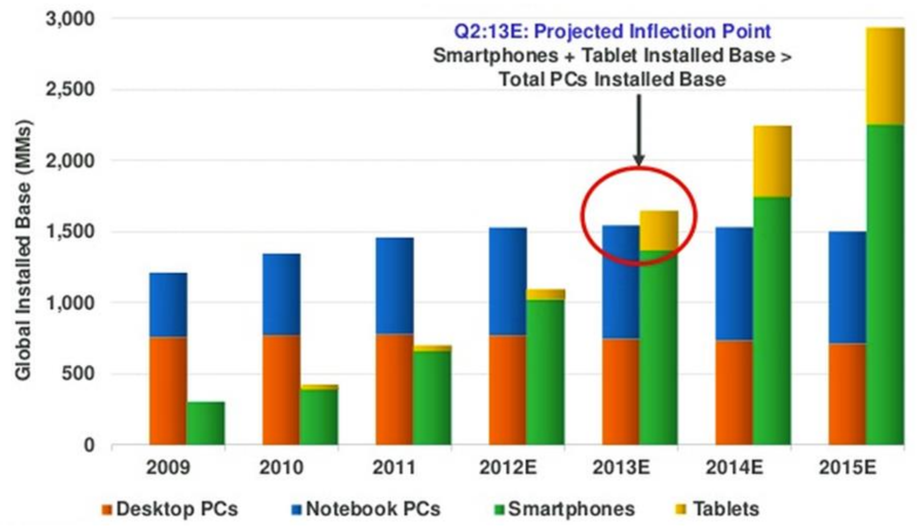


Figure 3: Global Installed Base of Desktop PCS + Notebooks PCS vs. Smartphones + Tablets

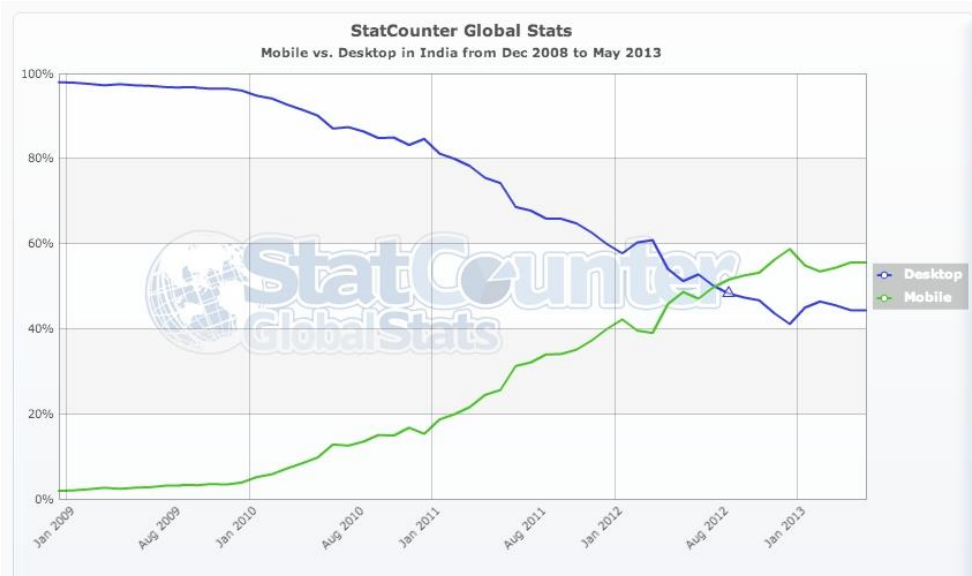


Figure 4: Mobile vs. Desktop in India from December 2008 to May 2013

Mobile devices have transformed the way that we interact and communicate. Tasks once handled by wired communication can now be performed by mobile devices equipped with several network interfaces (see Figure 5), which may be of different access technologies, both wireless and cellular ones. With such diversity, the fundamental goal [6] is to render transparent the existence of heterogeneous networks. Furthermore, selecting efficiently the most appropriate technology to use is crucial for obtaining the levels of performance required by future networks.

Figure 5: Mobile device equipped with several network interfaces

1.1 MOTIVATION

For some tasks, such as the ones involved during emergency network scenarios, the use of wireless devices is mandatory. Some relevant scenarios include coalition military operation, disaster relief efforts, and on-the-fly team formation for a common mission, such as search and rescue. In these situations, multiple groups and organizations may need to establish a way to communicate and collaborate to achieve a goal. For example in a disaster relief effort, a military force may need to coordinate its activities with fire fighters, medical teams, police forces and other entities by sharing information without being concerned with the particular networking technologies that each group uses.

Such tasks call for the development of an approach that enables end-to-end communications over those mobile wireless networks (i.e. networks containing wireless devices). The fundamental goal of mobile ad-hoc networking is to support efficient operation in these networks by incorporating routing functionality into mobile nodes, and into any device (i.e. router or end-system) that implements the internet protocol (IP) [7], [8]. A mobile ad-hoc network (MANET) is a multi-hop ad-hoc wireless network where nodes can move in an arbitrary manner in the topology. Therefore, the network may experience rapid and unpredictable topology changes. Such networks have no given infrastructure, can be set up quickly in any environment, and are likely composed of nodes with constrained capabilities (e.g. power level, processing capacity, and so forth). Moreover, this kind of network could be linked to other infrastructure networks constituting a mesh network.

Several MANET routing protocols have been specified by the IETF MANET WG and other entities [9] to achieve an easy deployment of these networks. These protocols are based on different design philosophies and propose to deal with certain requirements from different network domains. However, a network domain not fully explored by those is the efficient MANET deployment using heterogeneous technologies.

In 2012, the Heterogeneous Technologies Routing (HTR) protocol, a routing protocol to interconnect devices in a heterogeneous ad hoc network environment, was designed and implemented on real mobile devices [10]. The HTR protocol is part of a co-operation between the Ericsson Research [11], the Networking and Telecommunication Research Group (GPRT) [12] and the "Ministério da Ciência e Tecnologia e Inovação" (MCTI) [13]. It creates an enclosed heterogeneous MANET [14], which can be set up quickly in diverse environments and can be composed of different communication technologies such as Bluetooth [15], Wi-Fi [16]–[20]. HTR also provides self-organizing support to bootstrap its nodes, through the self-configuration of network interfaces requiring minimum human interaction. For energy awareness, HTR employs the HTRScore special metric to help the path computation process and interface selection. This is essential to mitigate excessive energy consumption since such networks are generally composed of nodes with constrained capabilities (e.g. energy level, processing capacity and so forth).

1.2 STATEMENT OF THE PROBLEM

The HTR protocol, in its baseline, only has support for real devices [21], [22] (see Figure 6) using Wi-Fi or Bluetooth devices. Since it currently only works using real devices, the setup of real experiments is extremely difficult to perform when validating large-scale networks. Furthermore, the HTR protocol is susceptible to routing loops that are associated to the HTRScore computation, which, in some cases, produces divergent views of the network since each node computes and propagates its own perception about the network energy consumption to reach others.

Figure 6: HTR prototype working devices.

1.3 PURPOSE OF THE RESEARCH

In this work, we develop a simulated version, using the network simulator 3 (ns-3) [23], of the HTR protocol in order to reduce the complexity of performing experiments that evaluate the performance of the protocol when using large-scale networks. We also extend the HTR protocol supported heterogeneous technologies with the addition of WiMAX [18] and LTE [24]–[27] devices. Furthermore, we propose an extension to enable multipath routing over that protocol in order to mitigate the routing loops occurring in HTR baseline. This thesis has the following specific objectives:

- Enhances the ns-3 energy component enabling it to compute the energy consumption of these two new heterogeneous technologies
- Investigates the impact of tuning routing parameters on convergence interval and energy consumption

Although a large number of works exist that investigate the tuning of routing parameter settings, to the best of our knowledge, none of them investigate the impact of these on protocol convergence and energy consumption.

1.4 RELEVANCY

This work performs a comparative analysis of the proposed HTR extension, with the baseline HTR, and the widely used Optimized Link State Routing (OLSR) [28] protocol. Moreover, we investigate the impacts of tuning the HELLO refresh interval and perform a comparative analysis of the tuned HTR (i.e. HTR with tuned routing parameter settings) with the OLSR protocol. Both evaluations are validated through the simulation of heterogeneous technologies such as WiMAX, LTE and Wi-Fi.

Contributions expected for this work are the following: first, a simulated version of the HTR protocol is given, providing an integrated solution for heterogeneous ad hoc communication scenarios, currently mostly lacking from industry and academia attention due to the high complexity acquired from the heterogeneous context. As a second contribution, effectively increase the network performance under the multipath extension. Third, investigate the impact of tuning routing parameters on the protocol convergence interval and energy consumption that, to the best of our knowledge, is not evaluated in other works. Further, discuss the impact of the heterogeneity, comparing the results obtained while using Wi-Fi, WiMAX and LTE technologies. Also, provide two new energy models for computation of energy consumption of WiMAX and LTE that enhance the ns-3 energy system which, at the moment of writing of this document, has only support for Wi-Fi devices.

1.5 DISSERTATION OUTLINE

This work has been organized as follows: In Chapter 2, background information is given with a review of current related works. Chapter 3 introduces the HTR protocol including its main concepts and features, and describes the multipath extension proposed here. Chapter 4 gives an introduction and evaluation of the network simulators, the ones that have gained attention in the research field, in order to discuss the reason for the use of the ns-3 network simulator during this work. Furthermore, the implementation of the HTR protocol in the ns-3 is presented and includes the system overview, UML diagrams, and the extensions developed to support the protocol itself. In Chapter 5, evaluations are performed considering a set of relevant metrics and heterogeneous scenarios to show and discuss the efficiency of the multipath extension as well as the tuning of routing parameter settings. Finally, concluding remarks and directions for future work are presented in Chapter 6.

CHAPTER 2 BACKGROUND AND RELATED WORK

In this chapter, an introduction to the some concepts regarding ad hoc network protocols is presented. Furthermore, there is a discussion of the related works.

2.1 BACKGROUND

An ad hoc network allows mobile hosts to access information and services, regardless of their geographic position. It is characterized by the randomness in its conformation and by the difficulties that imply performing routing tasks in an uncertain, dynamic and fast changing network environment.

MANET is an ad hoc network of nodes, which are connected by wireless links. There is no infrastructure and central command for communicating and controlling those nodes. It is a self-configuring network: the wireless nodes create the network among them. The nodes are always moving and they are working often as routers at the same time.

To provide communication between the nodes contained in a MANET, a routing process is used. Routing is the process of selecting paths in a network along which to send data or physical traffic. Routing directs the passing of logically addressed packets from their source toward their ultimate destination through intermediary nodes. Thus, routing protocol is the routing of packets based on the defined rules and regulations. Every routing protocol has its own algorithm on the basis of which it discovers and maintains the route.

To compare and analyze MANET routing protocols, appropriate classification methods are important [29], [30]. Several MANET routing protocols have been developed and can be classified according to different criteria. In [29], [31], a state-of-the-art review and a set of classification criteria for typical representatives of mobile ad hoc routing protocols are presented. These criteria are presented in the following subsection.

2.1.1 CHARACTERISTICS OF AD HOC ROUTING PROTOCOLS

Proactive, reactive and hybrid routing

Using a proactive routing protocol, nodes in a mobile ad hoc network continuously evaluate routes to all reachable nodes and attempt to maintain consistent, up-to-date routing information. Therefore, a source node can get a routing path immediately when it needs one. In addition, all nodes need to maintain a consistent view of the network topology. When a network topology change occurs, respective updates must be propagated throughout the network to notify the nodes of the change.

In a reactive routing protocol, also called an "on-demand" routing protocol, paths are searched only when needed. A route discovery operation invokes a route-determination procedure. The discovery procedure terminates either when a route has been found or when no route is available after examination of all route permutations.

Hybrid routing protocols are proposed to combine the merits of both proactive and reactive routing protocols and to overcome their shortcomings.

Structuring and delegating the routing tasks

Another classification method is based on the role that nodes may have. In a uniform routing protocol, all mobile nodes have the same role, importance, and functionality. This class of protocols normally assumes a plane structure. In a non-uniform routing protocol, some nodes carry distinct management and/or routing functions.

Exploring network metrics for routing

Some routing protocols utilize some metrics for routing path construction. These metrics can be used as criteria for mobile ad hoc network routing protocol classification. Most routing protocols use "hop number" as a metric [29]. If there are multiple routing paths available, the path with the lowest hop number will be selected.

Evaluating topology, destination and location for routing

In a topology based routing protocol, nodes collect network topology information for constructing routing decisions. In contrast, there are some destination-based routing protocols, in which a node only needs to know the next hop along the routing path when forwarding a packet to the destination.

The availability of GPS or similar location systems allows mobile nodes to access geographical information easily. In location-based routing protocols, the position relationship between a packet forwarding node and the destination can be used in both route discovery and packet forwarding.

Finally, for some mobile applications it is desirable to use the content of a message for routing instead of a destination address. Content-based routing conveniently supports these types of applications.

Protocols unicast and multicast

This classification is related to the capacity that the protocol has to send packets in a multicast or unicast manner.

Single-path and multipath

This classification is related to the number of paths computed during the path computation process. Unlike the single path strategy, with a multipath approach, different paths are computed between source and destination.

2.2 RELATED WORKS

The next subsections discuss the related works. First, the OLSR and OLSRv2 are introduced, The HTR protocol borrow some concepts from both protocols (OLSR and OLSRv2). Second, some approaches dealing with interoperability among heterogeneous networks are presented. Finally, some studies that investigate the tuning of routing protocol parameters and some solutions that use a multipath scheme are given.

2.2.1 OLSR AND OLSRV2

The OLSR was designed to work in a MANET context. OLSR is pro-active, unicast, link-state, table-driven and also uses a mechanism to control flooding (multipoint relaying - MPR) [32]. In a link state routing protocol, each node periodically broadcasts the current status of its links to all routers in the network. Whenever a link state change occurs, the respective notifications will be flooded throughout the whole network.

In OLSR, only nodes, selected as MPRs, are responsible for forwarding control traffic, intended for diffusion into the entire network. OLSR utilizes HELLO and Topology Control (TC) messages to discover and then disseminate link state information throughout the mobile ad hoc network. Individual nodes use this topology information to compute next hop destinations for all nodes in the network using shortest hop forwarding paths.

The OLSR version 2 (OLSRv2) [33]–[36] has the same algorithm and ideas as the first version (i.e. OLSR). Being modular by design, OLSRv2 is made up of a number of generalized building blocks, standardized independently and applicable also for other MANET protocols. Currently, these building blocks are, to the best of our knowledge, the following: the RFC 5148 “Jitter Considerations in Mobile Ad Hoc Networks (MANET)” [37], the RFC 5444 “Generalized Mobile Ad Hoc Network (MANET) Packet / Message Format” [38], the RFC 5497 “Representing Multi-Value Time in Mobile Ad hoc Networks (MANETs)” [39], the RFC 6622 “Integrity Check Value and Timestamp TLV Definitions for Mobile Ad Hoc Networks (MANETs)” [40], and the RFC 6130 “Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)” [41]. In addition to these building blocks, the OLSRv2 includes the updates of these RFC and the RFC of the OLSRv2 itself, which is in the final phases of standardization [42]. The [37], which is already considered in HTR protocol, provides recommendations for jittering (i.e. randomly modifying timing) of control traffic transmissions in MANET routing protocols in order to reduce the probability of transmission collisions. [38] specifies the syntax of a packet format designed for carrying multiple routing protocol messages for information exchange between MANET routers. [39] describes a general and flexible structure called TLV for representing time-values, such as an interval or a duration, and [40] proposes enhancements to include cryptography integrity check values (ICV) (i.e. digital signatures or MACs) in TLVs, both using the [38] format. Finally, [41] describes a 1-hop and symmetric 2-hop neighborhood discovery protocol for mobile ad hoc networks, which is quite similar to the link sensing method provided on OLSR. Actually, these new concepts introduced in OLSRv2 are been considered for inclusion into the HTR protocol extension and this is seen as a future work.

2.2.2 HETEROGENEOUS AD HOC NETWORKS

Recently, a number of approaches dealing with interoperability among heterogeneous ad hoc networks have emerged. While some have focused on the interoperation among multiple wireless domains, adopting high level architectures, and having merely sketched the required components (e.g. the translation of different naming spaces) [43]–[46], others have addressed the heterogeneous routing below the IP layer (underlay level) [47]–[49].

Inter-Domain Routing for MANETs (IDRM) [43], [44] enable end-to-end communications over heterogeneous MANETs governed by distinct administrative domains. It employs a path vector routing protocol, where each domain enumerates the entire domain level path to a destination. Using a path vector routing protocol, IDRM can potentially support a policy-based routing in a similar manner to BGP if the network topology is relatively stable. Dedicated gateway nodes are required to serve as bridges to connect heterogeneous MANETs networks.

Plutarch [45] is an architecture that translates address spaces and transports protocols among MANETs to support interoperation of heterogeneous networks, and TurfNet [46] is a proposal for inter-MANET networking without requiring global network addressing or a common network protocol. However, these are very high level architectures providing only sketches of the required components (e.g., translation of different naming spaces, and different protocols).

Ana4 [47] defines a generic layer 2.5 as part of an ad hoc architecture, which relies on the concept of a virtual ad hoc interface. This interface is a logical entity, which abstracts a set of network devices into a single and addressable network component. By designing an ad hoc proposal at layer 2.5, it becomes possible to provide end-to-end communication, regardless of the number of network interfaces in each node. In spite of the similarity with the proposed HTR protocol, Ana4 does not offer routing based on context information (e.g. residual energy), nor does it support MANET auto-organization.

Lilith [48] is an interconnection architecture based on label-switching for spontaneous edge networks, the main goal is to support TCP/IP applications without additional configuration from the user. To achieve this, Lilith makes use of an approach based on Multiprotocol Label Switching (MPLS) [50] to permit forwarding of packets over various heterogeneous links. A flow follows a Label Switched Path (LSP) established on demand by an ad hoc routing protocol. Lilith lacks support for the handling of logical sub-networks and self-configuration mechanism.

The 3D-Routing protocol [49] makes it possible to compose a fully connected heterogeneous MANET using a 2.5 layer approach. An interesting 3D-Routing concept is the use of roles. The idea is to allow nodes to have one or more roles associated with them. Despite its advantages, 3D-Routing introduces considerable overhead, increasing network bandwidth usage. The reasons for this are two-fold: the lack of a mechanism to control packet flooding, as well as the use of a complex representation scheme for node information and policies. Moreover, it does not describe the mechanism used to compute its routing table.

The Distributed Virtual Network Interfaces (DVNI) [51] put forward a solution in the context of Personal Area Networks (PANs) to provide heterogeneous devices and nodes the ability of discovering, seamless handover and routing capabilities. Their goal is to provide the so-called “Always Best Connected (ABC)” solution by employing a Virtual Network Interface (VNI) to address the issue of changing IPs due to mobility.

2.2.3 MULTIPATH ROUTING PROTOCOLS FOR AD HOC NETWORKS

Several multipath routing protocols were proposed for ad hoc networks [30]. Most of them are based on existing single-path ad hoc routing protocols. In [52], a multipath extension to the well-known AODV [53] protocol is introduced. In [54], a new QoS-aware multipath source routing protocol called MP-DSR, and based on DSR [55], is designed and it focuses on a new QoS metric to provide increased stability and reliability of routes. [56] introduces a multipath Dijkstra-based algorithm to obtain multiple routes and it uses a proactive routing protocol based on OLSR [28]. [57] recommends a scheme that uses two disjoint routes for each session using an on-demand approach.

Notwithstanding the important contributions of these existing solutions, there is still a need for an efficient multipath scheme to connect heterogeneous devices seamlessly.

2.2.4 *PARAMETER BASED STUDIES*

Over the years, some studies were done in order to increase the performance of OLSR [28]. In [58], the author investigates the different impacts of tuning the refresh interval timers on OLSR performance under various scenarios. This work proves that the intervals for HELLO messages have a bigger impact on OLSR performance than other message types. [59] also proposed to evaluate the OLSR protocol with different route refresh intervals. However, in both cases, the performance was measured basically in terms of the network metrics' throughput and delivery ratio, while the convergence and energy efficiency of the network were not taken into consideration.

Meanwhile, other works [60]–[62] explore the accuracy of state information while tuning the OLSR parameters and conclude that changing parameters has no noticeable impact on accuracy levels. In addition, [60] considers the network heterogeneity by using radios with distinct power consumptions and with different initial energy levels; however, it does not assume the use of heterogeneous wireless technologies.

In [63], [64], the authors experiment with a set of OLSR settings using a real network environment and investigate the impact on energy consumption, throughput, and route change latency. They conclude that route change latency may be reduced with the use of adequate protocol parameter settings; however, only three possible setups were evaluated, leading to some minor improvement of the battery lifetime.

In [65], the author presents a performance comparison of OLSR and *Ad hoc* On-Demand *Distance Vector* (AODV) [53] using default and modified parameters in order to achieve better transmission; however, it confirmed that, although the modification of selected routing parameters may be beneficial, it can bring deterioration of transfer characteristics for some network topologies with a specific mobility level. In [66], the author evaluated the performance of the AODV routing protocol when using IEEE 802.15.4 sensor devices, by changing the parameters associated with the algorithm in order to enhance performance in terms of throughput, end-to-end delay, energy consumption, and packet delivery ratio.

Notwithstanding the important contributions of the existing solutions, none of them investigates the effects of tuning protocol parameters under scenarios that use heterogeneous technologies such as WiMAX, Wi-Fi and LTE. Additionally, none of them discusses the impact of this heterogeneity on routing protocol convergence nor do they consider the energy consumption during this stage.

2.3 CHAPTER SUMMARY

In this chapter, we introduced some concepts of MANET and introduced the OLSR (version 1 and 2) that is the protocol that the HTR protocol is based. Furthermore, we discuss some works dealing with interoperability among heterogeneous networks, some studies that investigate the tuning of routing protocol parameters and some solutions that use a multipath scheme.

CHAPTER 3 HTR PROTOCOL

The HTR is a routing protocol for heterogeneous mobile ad hoc networks, which abstracts multi-hop, multiple and heterogeneous interfaces into a single ad hoc network. HTR offers a proactive cross layer routing protocol, complete TCP/IP compatibility, and a logical mechanism for network partitioning and merging. Being proactive, HTR ensures that a path is computed as early as possible. HTR provides a unique IP address for each network device, regardless of the number and types of interfaces; this address will remain the same for as long as the device is connected, despite high mobility. To guarantee address uniqueness in an evolving environment, HTR adopts the Network Address Allocation Method proposed in [67].

The soft state signaling approach [68] is used in HTR. This protocol has two periodic control message types: HELLO and Topology Control (TC). Periodic refresh messages make the system robust against node failure due to loss/corruption of refresh messages, and there is no requirement for guaranteed delivery of refresh messages. The internal state maintenance in each node is related directly to the refresh intervals and so changing these will impact the protocol as a whole. HTR is based on the OLSR protocol and includes MPR for reduction of traffic flooding. From a HELLO message, a mobile node receives information about its immediate, 2-hop neighbors and selects MPRs accordingly. A TC message originates at an MPR node announcing who has selected it as an MPR. In contrast to OLSR, HTR uses additional metrics such as the awareness of link conditions and power efficiency, in order to choose MPR nodes and path computation. Furthermore, HTR uses control messages to piggyback service propagation and policies that are based on human roles; however, these last two features are not provided in this work as they are seen as an extension for future works.

3.1 HTR TERMINOLOGY

Node: A MANET device which implements the HTR protocol as specified in this work.

HTR interface: Each interface of a network device which may have several interfaces, although the HTR attributes only one, unique, IP address to the whole node.

Neighbor node (1-hop neighbor): A node which is a 1-hop neighbor of a given local node, if there is any local interface that has a direct link to at least one interface of this neighbor.

2-hop neighbor: A node which is two hops away from another node and which can be reached (in terms of radio range) directly from at least one 1-hop neighbor.

Multipoint Relay (MPR): A node which is selected by a 1-hop neighbor, node X, as the forwarding node to retransmit a broadcast packet received from X if this one is not duplicated and has a "time to live" field greater than one.

MPR Set: A (sub)set of neighbors selected in such a way that it covers (in terms of radio range) all symmetric strict 2-hop neighbors.

Multipoint Relay selector (MPR selector, MS): A node which has selected its 1-hop neighbor as an MPR.

Link: A node which is said to have a link to another node when one of its interfaces has a link to one of the interfaces of another node.

Symmetric link: A verified bi-directional link between two HTR interfaces.

Asymmetric link: A link between two HTR interfaces, verified in only one direction.

Symmetric 1-hop neighborhood: The symmetric 1-hop neighborhood of any node X.

Symmetric 2-hop neighborhood: The symmetric 2-hop neighborhood of X.

The Figure 7 illustrates some terminologies presented along this section.

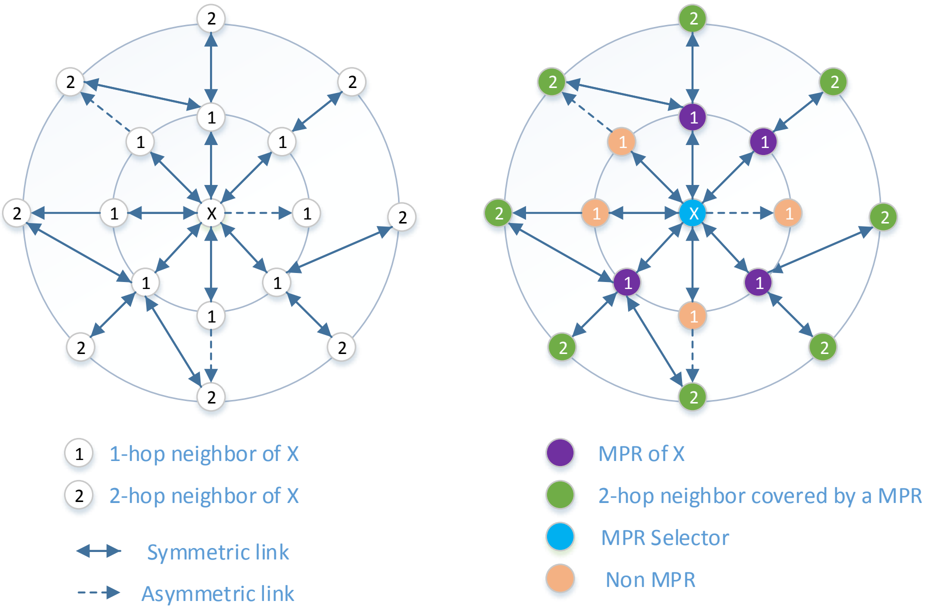


Figure 7: Terminology

3.2 FUNCTIONALITIES

The HTR protocol operates according to a set of well-defined functionalities in order to enable the routing in stand-alone MANETs. These functions specify the behavior of nodes participating in the MANET and running HTR as their routing protocol. The main functionalities of HTR are:

Node addressing: designed to operate on nodes with multiple communication interfaces, this functionality attributes a unique IP address to identify a node. HTR can use IP version 4 (IPv4) [7] and version 6 (IPv6) [8].

Link sensing: this functionality keeps up-to-date information on what links exist between a node and its neighbors. The connectivity is checked through periodic emission of HELLO (heart beat) messages over the interfaces. A separate HELLO message is generated for each interface and emitted in correspondence with the provisions discussed in Section 3.7. Link sensing information is stored in the Link-Neighbor set repository that will be explained in Section 3.5.

Neighbor discovery: this functionality allows nodes to discover the existence of neighbors through the information exchanged during link sensing. Neighbor information is stored in the Link-Neighbor set repository.

MPR selection and MPR signaling: the objective of MPR selection is for a node to select a subset of its 1-hop neighbors as the forwarding node set to retransmit broadcast packets. Other nodes that are not "on" in the MPR set can process but not retransmit the broadcast packet. The MPR set guarantees that all 2-hop neighbors of each node receive a copy of the packet and, therefore, all nodes in the network can be covered without retransmissions by every node. The MPR selection and signaling control the network flooding by minimizing redundant retransmissions locally. The process of MPR selection is detailed in Section 3.8. MPR signaling refers to the process of propagating the MPR set computed and is discussed in Section 3.7.

Topology Control message diffusion: Topology Control messages are broadcast to provide each node with sufficient link-state information to allow route calculation. Topology Control messages diffusion is described in Section 3.7.

Path computation: allows each node to compute its routing table according to link quality information obtained through periodic message exchanges. Section 3.9 and 3.10 details this functionality.

3.3 NODE ADDRESSING

Since the address allocation and maintenance in MANET is not a trivial task, the HTR protocol uses, for automatic address organization, a protocol named Dynamic Node Configuration Protocol (DNCP) [69]. The DNCP employs a distributed generation function, defined in [67], and also techniques to overcome the challenges inherited from the dynamic behavior of MANET. The main parts of DNCP can be divided in two: address allocation and address maintenance. The first one allocates a unique identifier for each node and the second one has to deal with network disconnections, recovery of lost addresses, and duplication that may arise when partition and network merges occurs. In [69], the author describes the DNCP and its components. The identification address generated using DNCP is converted to an IP address (version 4 or version 6) that is used to identify each node contained in the HTR network.

3.4 HTRSCORE

One of the differences between HTR and other protocols is its engagement to keep the network operating as long as possible. Many routing approaches use only the hop count during routing path computation [29]. In contrast, HTR uses a cost metric called HTRScore that is defined using factors such as the awareness of link conditions and power efficiency in order to perform path computation and which is also applicable for MPR set computation. HTRScore also introduces the packet loss probability, a factor that evaluates link stability, an optimization that benefits paths with high packet delivery. The adopted HTRScore formula can be seen at (1).

$$HTRScore\ (i,j) = \frac{e_{i,j}^{\alpha}}{(1-\rho_{i,j})^{\beta}} . \frac{E_i^{\gamma}}{R_i^{\theta}} \tag{1}$$

where i is the source node; j is the destination neighbor; $e_{i,j}$ is the transmission energy required for node i to transmit an information unit to its neighbors j; $\rho_{i,j}$ is the probability to lose a packet sent from i to j; R_i is the residual energy of node i; and E_i is the initial battery energy of node i.

The parameters α, β, γ, and θ represent nonnegative weighting factors for each described parameter. Note that, if all weights are equal to zero, then the lowest-cost path is the shortest path, and if only γ and θ are equal to zero, then the lowest cost path is the one that will require less energy consumption, independently of whether retransmission is considered or not, regarding the value of β. If γ is equal to θ then normalized residual energy is used, while if only θ is equal to zero then the absolute residual energy is used. When α, γ and θ are equal to zero, then only the paths with best link stability are emphasized.

3.5 INFORMATION REPOSITORIES

Since HTR is a pro-active and table-driven protocol which stems from a conventional link state routing algorithm, it maintains information repositories (databases), which represent network topology. These information repositories are updated regularly via control messages in order to maintain up-to-date routing information. Below, a brief explanation of each repository is presented.

3.5.1 LINK-NEIGHBOR SET BASE

The **Link-Neighbor Set Base** repository is responsible for maintaining the state of 1-hop neighbors. It is important to emphasize that each register represents a single link established between two interfaces, both representing the link communication from one node to its 1-hop neighbor. Basically, seven fields compose each register of this repository:

Local Interface Identifier (L_Iface_ID): represents the local interface index, in which the link is established with the 1-hop neighbor (16 bits). This interface index is set by the autoconfiguration process (i.e. DNCP) and represents the interface index used as interface identification on the HTR protocol.

Neighbor Address (N_Addr): represents the IP address of the 1-hop neighbor node (32 bits for IPv4 and 128 bits for IPv6).

Status: indicates whether the link is symmetric or not (1 bit). The value 0 represents a non-symmetric link (**ASYM**) and a value 1 represents a symmetric link (**SYM**). This field is set during the link sensing and neighbor discovery process, defined in Section 3.7.

SYM Time: indicates the amount of time in seconds that the link is still considered as symmetric (8 bits). Whenever the interval expires, the link is declared as asymmetric. This timer can be updated (i.e. reset) if the neighbor discovery process detects that the link is still established as symmetric.

ASYM Time: indicates the amount of time in seconds that the link is still valid and not considered as lost. During this interval, the link is considered as an asymmetric link; whenever the interval expires the link is considered as lost, if not updated (i.e. reset) by the neighbor discovery process. (8 bits).

Time: stores the validation time (expiration time) in seconds of the record. This time specifies the time at which the record expires and must be removed. This interval is used to remove lost links (i.e. remove the record when the SYM Time and ASYM Time expire). (8 bits).

HTRScore: represents the energy awareness metric used to compute MPR set and path computation (32 bits). This metric represents the cost to transmit to the 1-hop neighbor.

The information of this repository is acquired via HELLO messages and is used to compute the MPR set that is stored in the MPR Set Base**, and to** generate HELLO messages and path computation. The Table 1 illustrates a sample of the Link-Neighbor Set Base repository.

Table 1: Link Neighbor Set Base

L_Iface ID	N_Addr	Status	SYM Time	ASYM Time	Time	HTRScore
1	192.168.0.4	0	0	30	30	120
2	192.168.1.26	1	15	0	30	30
3	192.168.0.4	1	30	0	30	80
1	192.168.0.9	0	0	10	10	40

3.5.2 TWO-HOP NEIGHBOR SET BASE

This repository maintains the set of 2-hop neighbors of the node. Each record in this repository represents a symmetric link between the 1-hop neighbor node and the 2-hop neighbor covered by this 1-hop neighbor. The 1-hop neighbor is registered in the Link-Neighbor Set Base. Basically, four fields compose each record of this repository:

Neighbor Address (N_Addr): represents the IP address of the 1-hop neighbor that covers the 2-hop neighbor (32 bits for IPv4 and 128 bits for IPv6).

Two-Hop Neighbor Address (2Hop_N_Addr): represents the IP address of the two-hop neighbor node (32 bits for IPv4 and 128 bits for IPv6).

Time: stores the validation time (expiration time) in seconds of the record (8 bits). Whenever this interval expires, the record must be removed.

HTRScore: represents the energy awareness metric used to compute MPR set and path computation (32 bits). This metric represents the cost to transmit to the 2-hop neighbor, computed and broadcasted by the 1-hop neighbor.

Similar to the Link-Neighbor Set Base, this repository is filled via HELLO messages and is also used to compute MPR set and path computation. The Table 2 illustrates a sample of the Two-Hop Neighbor Set Base repository.

Table 2: Two-Hop Neighbor Set Base

N_Addr	2Hop_N_Addr	Time	HTRScore
192.168.0.4	192.168.0.5	10	20
192.168.1.26	192.168.0.15	15	100
192.168.0.4	192.168.1.25	30	80
192.168.0.9	192.168.0.26	0	40

3.5.3 MPR Set Base

The MPR Set Base contains the set of 1-hop neighbors selected as MPR. The MPR optimization will be described afterward in Section 3.8. Basically, two fields compose each record of this repository:

Neighbor Address (N_Addr): represents the IP address of the MPR neighbor (32 bits for IPv4 and 128 bits for IPv6).

Time: stores the validation time (expiration time) in seconds of the record (8 bits).

This repository is computed by the MPR computation process that uses information from the Link-Neighbor Set Base and the Two-Hop Neighbor Set Base repositories. The information provided here is also used during the generating of HELLO messages, considering that each node broadcasts its MPR set. The Table 3 gives a sample for the MPR Set Base.

Table 3: MPR Set Base

N_Addr	Time
192.168.0.4	20
192.168.0.9	20

3.5.4 MPR Selector Set Base

The MPR Selector Set Base contains all neighbor nodes that have selected the node as MPR. Its structure is quite similar to the MPR Set Base, where two fields compose each record of the repository:

Neighbor Address (N_Addr): represents the IP address of the 1-hop neighbor that selected the node as MPR (32 bits for IPv4 and 128 bits for IPv6).

Time: stores the validation time (expiration time) in seconds of the record (8 bits).

The information to fill this repository is acquired via HELLO messages and is used to generate TC messages and to forward HTR messages, since only selected MPRs can forward broadcasted messages. Table 4 gives an example of the MPR Selector Set Base.

Table 4: MPR Selector Set Base

N_Addr	Time
192.168.0.4	20
192.168.0.9	20

3.5.5 TOPOLOGY CONTROL BASE

The Topology Control Base repository is responsible for maintaining topology information about the network. Basically, five fields compose each record of this repository:

TC Destination Address (TC_Dest_Addr): the destination address of the node (32 bits for IPv4and 128 bits for IPv6) that may be reached in one hop from the node with the address **TC_Last_Addr**.

TC Last Address (TC_Last_Addr): the address (32 bits for IPv4and 128 bits for IPv6) of the MPR of **TC_Dest_addr**.

Sequence Number (Seq): sequence number used to keep track of the most recent topology information (16 bits).

HTRScore: represents the energy awareness metric used to compute MPR set and path computation. This metric represents the cost to transmit from TC_Last_Addr to TC_Dest_Addr.

Time: stores the validation time (expiration time) in seconds of the record.

The information in this repository is acquired via TC messages and used for path computation. The Table 5 illustrates a sample of the Topology Control Base repository.

Table 5: Topology Control Base

TC_Dest_Addr	TC_Last_Addr	Seq	Time	HTRScore
192.168.0.80	192.168.0.4	101	15	90
192.168.1.15	192.168.1.26	100	10	200
192.168.0.20	192.168.0.4	104	30	30
192.168.0.70	192.168.0.9	100	10	50

3.5.6 DUPLICATE BASE

This repository maintains all received and processed messages. Basically, five fields compose each record of this repository:

Duplicate Address (D_Addr): the originator address (32 bits for IPv4 and 128 bits for IPv6)of the message.

Sequence Number (Seq): sequence number of the message (16 bits).

Duplicate Retransmitted (D_Retr): indicates whether the message has been already retransmitted (1 bit). Typically, this field is represented by a bit value, where 0 represents that the message has not already been relayed (FALSE) and 1 represents that the message was already relayed (TRUE).

Duplicate Interface List (D_Iface_List): is the list containing the index of each interface on which the message has been received.

Time: stores the validation time (expiration time) in seconds of the record.

The information to fill this repository is acquired via HELLO and TC messages and is used to disallow the forwarding of duplicate HTR messages. The Table 6 uses a set of records to illustrate the Duplicate Base**.**

Table 6: Duplicate Base

D_Addr	Seq	D_Retransmitted	D_Iface_List	Time
192.168.0.80	100	TRUE	{1}	30
192.168.1.15	101	FALSE	{1,2}	20
192.168.0.20	102	TRUE	{1}	30

3.6 PACKET FORMAT AND FORWARDING

This section describes the format of the HTR packet, the packet processing and forwarding algorithm, and as well discusses the jitter considerations introduced in [37].

3.6.1 PACKET FORMAT

The basic layout of the HTR packet is shown in Figure 8. The HTR packet can contain one or more messages (e.g. HELLO and TC) per packet, as illustrated below.

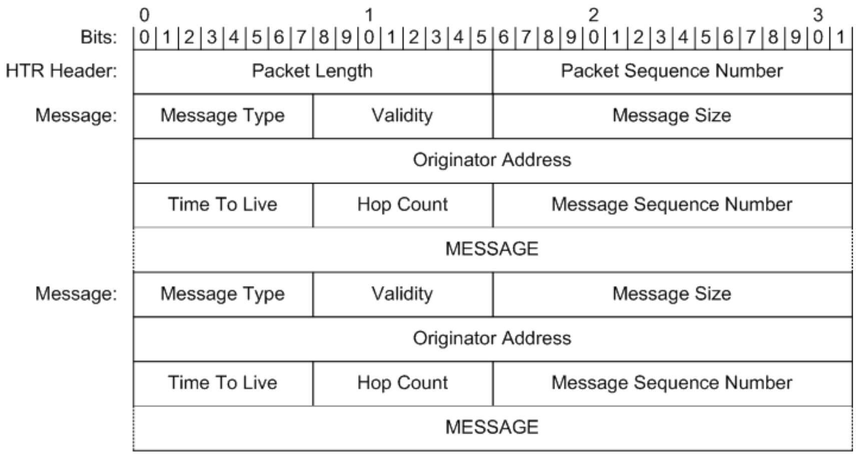


Figure 8: HTR packet format

HTR packet header:

Packet Length: the length (in bytes) of the entire packet, including the header size.

Packet Sequence Number: a sequence number incremented by one each time a new HTR packet has to be transmitted by the host (16 bits). The Packet Sequence Number is used to identify the occurrence of packets that were already processed (i.e. duplicate packet).

HTR message header:

Message Type: an integer identifying the type of message (8 bits). Similar to OLSR, HTR reserves the 0-127 space for message types, while the 128-255 space is reserved for future extensions.

Validity: indicates for how long, after reception, a node considers valid the information contained in the message (8 bits). The time interval is represented in a mantissa-exponent format [70].

Message Size: indicates the size (in bytes) of the message, including message header (16 bits).

Originator Address: indicates the address of the node that has originally generated the message (32 bits). It is important to emphasize that this field cannot be mixed up with the source address from the IP header, which is changed each time that the message is re-transmitted to the address of the relaying node.

Time To Live (TTL): an integer identifying the maximum number of hops this message can be forwarded to (8 bits). It enables flooding control. For each retransmission of the message, this field is decremented by 1. A node that receives a message with a TTL less than or equal to 1 cannot retransmit it.

Hop Count: an integer identifying the number of times the message has been forwarded (8 bits). Initially, this field is set to 0 and after each retransmission it is incremented by 1.

Message Sequence Number: a sequence number incremented by 1each time a new HTR message is transmitted (16 bits). The sequence number prevents the retransmission of messages sent previously.

The Message field contains the message (for example the HELLO and TC messages) which is described in Section 3.7.

3.6.2 Packet Processing and Message Flooding

The packet processing starts with the evaluation of the packet header to determine if any message exists and has to be processed. After, the message header for each message is evaluated and the value of the Message Type field is checked to determine the type of message so that the protocol knows how to process it. Since a node may receive a large number of messages in a short time, there is the possibility that some already processed message can be received and processed again. To avoid this waste of resources, HTR nodes record information about the most recently received messages in Duplicate Base.

For each received HTR packet, the following conditions apply:

Packet dropping condition:

1. If the packet contains no messages (i.e. the Packet Length has the size of the packet header only), the packet is discarded.
2. If the packet was sent by the receiving node, the packet is discarded.

For each message contained in the packet:

Message processing condition:

1. If there is a tuple in the Duplicate database, which satisfies the following condition:

If D_Addr is equal to the Originator Address and Seq is equal to the Message Sequence Number, then the message must not be processed since it was previously processed (i.e. duplicated message).Otherwise, the message is processed according to the specifications for the message type.

2. If the value of the TTL of the message is equal to zero, the message must not be processed.

Message forwarding condition:

The message is retransmitted using the default forwarding algorithm.

3.6.3 DEFAULT FORWARDING ALGORITHM

The default forwarding algorithm operates as follow:

1. If the sender of the message is not detected as a symmetric 1-hop neighbor, then the forwarding algorithm stops and the message is not forwarded.
2. If there is a tuple in the Duplicate database that satisfies the following condition: If D_Addr is equal to the Originator Address and Seq is equal to the Message Sequence Number, then the message will be forwarded if and only if the D_retr field of the Duplicate base tuple is equal to FALSE and the index of the interface which received the message is not included among the interfaces included in D_Iface_List.
3. If an entry doesn't exist in the Duplicate database, the message can be forwarded.

If, after those steps, the message is not considered for forwarding, the processing stops (i.e. steps 4 to 8 are ignored). Otherwise, the following algorithm is used:

4. If the sender is an MPR selector of this node and if the time to live of the message is greater than 1, the message is retransmitted.
5. If a tuple in the Duplicate database exists, with the same Originator Address, and the same Message Sequence Number, the tuple is updated as follows:
    a. Time is set to DUP_HOLD_TIME. The DUP_HOLD_TIME is the Validity time for the tuples recorded in the Duplicate Set and has a value of 30 seconds for the HTR.
    b. The receiving interface index (i.e. interface which received the message) is added to D_Iface_List.
    c. D_Retr is set to TRUE.

6. Otherwise an entry in the Duplicate database is created with the following fields:
    a. D_Addr is set to the Originator Address.
    b. Seq is set to the Message Sequence Number.
    c. Time is set to the DUP_HOLD_TIME.
    d. D_Iface_List, initially empty, is set to contain the index of the receiving interface.
    e. D_Retr is set to TRUE.
7. The TTL field of the message is reduced by one.
8. The hop-count of the message is increased by one.

    The message is broadcast on all interfaces (retransmission).

3.6.4 MESSAGE EMISSION AND JITTER

Synchronized transmissions may lead to packet collision due to simultaneous transmission. To avoid collisions, a random small time interval for which the message has to be cached before it is transmitted, known as jitter, is introduced. To reduce the likelihood of collisions or, in case of occurrence, prevent it from continuing, three jitter mechanisms can be applied for different cases:

- Periodic message generation.
- Externally triggered message generation.
- Message forwarding.

In the first case, jitter reduces the interval between successive message transmissions by a random time. For the latter two cases, jitter is used to delay a message being generated or forwarded by a random time. Each of these cases is described individually below and each of them uses a parameter, denoted MAXJITTER, for the maximum timing variation that it introduces. Some points with relevance of how the maximum jitter (MAXJITTER) may be determined are given below in subsection 3.6.8.

3.6.5 PERIODIC MESSAGE GENERATION

If two nodes generate packets containing regularly scheduled messages of the same type at the same time, and if, as typical, they are using the same message interval, all further transmissions of these messages will also be at the same time.

When a node generates a message periodically, two successive messages will be separated by a well-defined interval, denoted MSG_INTERVAL. A node may maintain more than one such interval, e.g., for different message types or in different circumstances (such as backing off transmissions to avoid congestion).

For this case, the jitter should be applied by reducing this delay by a random amount, so that the delay between consecutive transmissions of messages of the same type is equal to (MSG_INTERVAL - jitter), where jitter is the random value. Subtraction of the random value from the message interval ensures that the message interval never exceeds MSG_INTERVAL. The jitter value should be generated uniformly in an interval between zero and MAXJITTER.

3.6.6 EXTERNALLY TRIGGERED MESSAGE GENERATION

If nodes respond to changes in their circumstances, in particular changes in their neighborhood, with an immediate message generation and transmission, then two nearby nodes that respond to the same change will transmit messages simultaneously. To reduce this likelihood, an externally triggered message should be jittered by delaying.

When messages are triggered, whether or not they are also periodically transmitted, the protocol may impose a minimum interval between messages of the same type, denoted MSG_MIN_INTERVAL.

In the case that such an interval is not required, MSG_MIN_INTERVAL is zero. When MSG_MIN_INTERVAL is non-zero, it is appropriate to allow this interval to be reduced by jitter. Thus, when a message is transmitted, the next message is allowed after an interval of (MSG_MIN_INTERVAL - jitter).The jitter should be generated uniformly in an interval between zero and MAXJITTER.

3.6.7 MESSAGE FORWARDING

If nodes forward messages received from other nodes, then nearby nodes will commonly receive and forward the same message. If forwarding is performed immediately, the resulting packet transmissions may interfere with each other. To reduce the interference, when a node forwards a message, the message should be jittered.

For several possible reasons (differing parameters, message rescheduling, extreme random values), a node may receive a message while still waiting to forward an earlier message originated by the same node. This is possible without jitter, but may occur more often with it. The appropriate action to take is to forward only the most recent message if the types are the same. Other action would be to forward both messages if the types are different. The grouping of messages is supported through the general HTR packet that conveys more than a message per packet and could also be applied.

3.6.8 MAXJITTER

The steps involved during the generation of MAXJITTER must follow some recommendations introduced by [37]and listed below:

- All of the following are relevant to be applied to each instance of MAXJITTER:
  - It must not be negative.
  - It must not be greater than MSG_INTERVAL/2.
  - It should not be greater than MSG_INTERVAL/4.
- If MSG_MIN_INTERVAL > 0, then:
  - MAXJITTER must not be greater than MSG_MIN_INTERVAL.
  - MAXJITTER should not be greater than MSG_MIN_INTERVAL/2.
- As well as the decision whether to use jitter being dependent on the medium access control and lower layers, the selection of the MAXJITTER parameter should be appropriate to those mechanisms. For example, MAXJITTER should be significantly greater than (i.e. an order of magnitude greater than) any medium access control frame period.
- As jitter is intended to reduce collisions, greater jitter, i.e. an increased value of MAXJITTER, is appropriate when the chance of collisions is higher. This is particularly the case with increased node density, which is significant relative to (the square of) the interference range rather than useful signal range.
- The choice of MAXJITTER used when forwarding messages may also take into account the expected number of times that the message may be sequentially forwarded, up to the network diameter in hops, so that the maximum accumulated delay is bounded.

The HTR MAXJITTER is defined as the MSG_INTERVAL/4.

3.7 MESSAGES TYPES AND GENERATION

HTR defines only two message types: HELLO and TC (Topology Control). All functionality of HTR is based on generation and processing of these two messages. However, the HTR protocol packet format allows for a wide variety of custom messages to be transmitted and flooded according to the needs of the designer. In addition, the MPR optimization used in the protocol makes for the possibility of efficient message flooding to anyone in need of net-wide broadcasting of traffic in the ad hoc network.

3.7.1 LINK SENSING AND NEIGHBOR DISCOVERY

The neighborhood discovery activity is responsible for discovering nodes which are directly reachable and able to communicate. Since neighborhood discovery is used for network topology construction and consequently used for path computation, its correctness and accuracy is essential. Typically, these messages are sent in regular intervals to verify whether or not the established communication remains active.

HTR protocol adopts the same mechanism to discover neighbors as used by OLSR, and Figure 9 illustrates the neighbor discovery process.

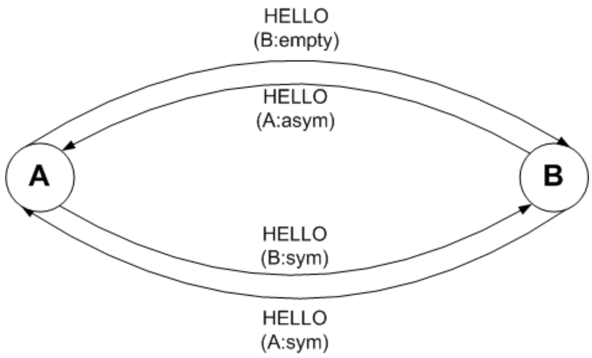


Figure 9: Neighbor discovery using only HELLO message

The standard algorithm for the neighbor discovery process via HELLO messages performs the following steps:

1. A node A that desires to discover its neighbors broadcasts a HELLO message containing its own neighbors, if any, and information regarding such neighbors, including link states and type (Link-Neighbor Set Base). If it does not have any neighbors, the HELLO is sent empty.
2. A node B receives the HELLO message sent by node A and could register it as an asymmetric or symmetric neighbor depending on the context. If node B finds its own address in the HELLO message, it will register A as symmetric. Otherwise, A is registered as asymmetric if B cannot find its own address in the HELLO message. Afterward, node B must include node A in its next HELLO message to be sent, since A is, at this moment, a neighbor of it.
3. When node A receives a HELLO message and finds its own address in it, it must set the node as a symmetric neighbor and include this node in the next HELLO message to be sent.

4. If the HELLO message sent by node B declares A as asymmetric, then node A includes node B in the next HELLO message to be sent. Thus node B, after the reception of the next HELLO message from node A, must register node A as a symmetric neighbor.

HELLO messages are broadcast to reach only one-hop neighbors and are used to perform link sensing and neighbor sensing, as well as to detect 2-hop neighbors that are deduced using the information provided by 1-hop neighbors (HELLO messages), and to broadcast the MPR computed set that is used to fill the MPR Selector base (MPR selector signaling). Therefore, the HELLO messages enable the HTR protocol to populate some of its repositories, specifically the Link-Neighbor Set Base, the Two-Hop Neighbor Set Base and the MPR Selector Base. Thus, HELLO messages support four essential tasks: Link sensing, Neighbor discovery, detection of 2-hop neighbors, and MPR selection signaling.

3.7.2 HELLO Message

HELLO messages are used to populate the majority of the repositories (Link-Neighbor Set, Two-Hop Neighbor Set, and MPR Selector Set) with the state of links and information about the known neighbors (e.g. address, type and energy awareness) of the 1-hop neighbor broadcasting the HELLO message. The broadcasting node, while generating the HELLO message, classifies and groups its known neighbors and sets the Link Code field using this classification. The groups classified are held in the Data field that exists for each Link Code. The HELLO message can hold more than one classified group, known as Link Message, if necessary. The fields contained in the HELLO message are described below in the Format subsection.

3.7.2.1 HELLO Format

The format of the HELLO message is as follows (Figure 10):

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Link Code								**Reserved**								**Link Code Size**															
Data																															
Link Code								**Reserved**								**Link Code Size**															
Data																															

Figure 10: HELLO message format

- **Link Code:** this field is the one to categorize the 1-hop neighbors known by the originator of the message (i.e. broadcasting node). This field describes the state of the link established by this originator to the set of known 1-hop neighbors contained in the Data field (i.e. the group of 1-hop neighbors known by the originator and classified relative to the Link Code) and as well as the type of this set (8 bits). The Link code can be divided into two blocks (2 bits each); one is the Link Type and the other the Neighbor Type. The Link Type indicates the state of the links, i.e. if the links established with the set, described by the Link Code, are symmetric, asymmetric, lost, or whether no information is known about the links. The Neighbor Type indicates the type of the set, and that includes whether the set is considered by the originator as symmetric or asymmetric neighbors, or if these are considered by the originator as its MPR set. The Link Code field is illustrated in Figure 11 and, as well, the possible states for each block (Link Type and Neighbor Type) are described.

7	6	5	4	3 2	1 0
0	0	0	0	Neighbor Type	Link Type

Figure 11: Link Code field format

The subfield Link Type can represent four distinct states:

00 (**UNSPEC_LINK**): indicates that no specific information about the link is given (reserved for future use).

01 (**ASYM_LINK**): indicates that the links are asymmetric.

10 (**SYM_LINK**): indicates that the links are symmetric.

11 (**LOST_LINK**): indicates that the links are considered as lost.

The subfield Neighbor Type can represent three distinct states:

00 (**SYM_NEIGH**): indicates that the neighbors have at least one symmetrical link with the originator.

01 (**MPR_NEIGH**): indicates that the neighbors have at least one symmetrical link and have been selected as MPR of the originator.

10 (**NOT_NEIGH**): indicates that the nodes are either no longer symmetric neighbors or have not yet become symmetric neighbors (i.e. asymmetric neighbor).

Reserved: a field reserved for future use (8 bits). It is filled with value integer "0".

Link Code Size: indicates the size (in bytes) of the link message, beginning from the Link Code field until the beginning of the next Link Code field or, if there is no other Link Code, the end of the message (16 bits).

Data: represents a variable field (Figure 12) composed by the 1-hop neighbors that includes the following information for each of these:

Neighbor IP Address: indicates the address of the 1-hop neighbor of the originator (32 bits for IPv4 and 128 bits for IPv6).

HTRScore: represents the energy awareness metric used to compute the MPR set and path computation (32 bits). This metric represents the cost to transmit from the originator to the 1-hop neighbor with address equal to the "Neighbor IP Address" field.

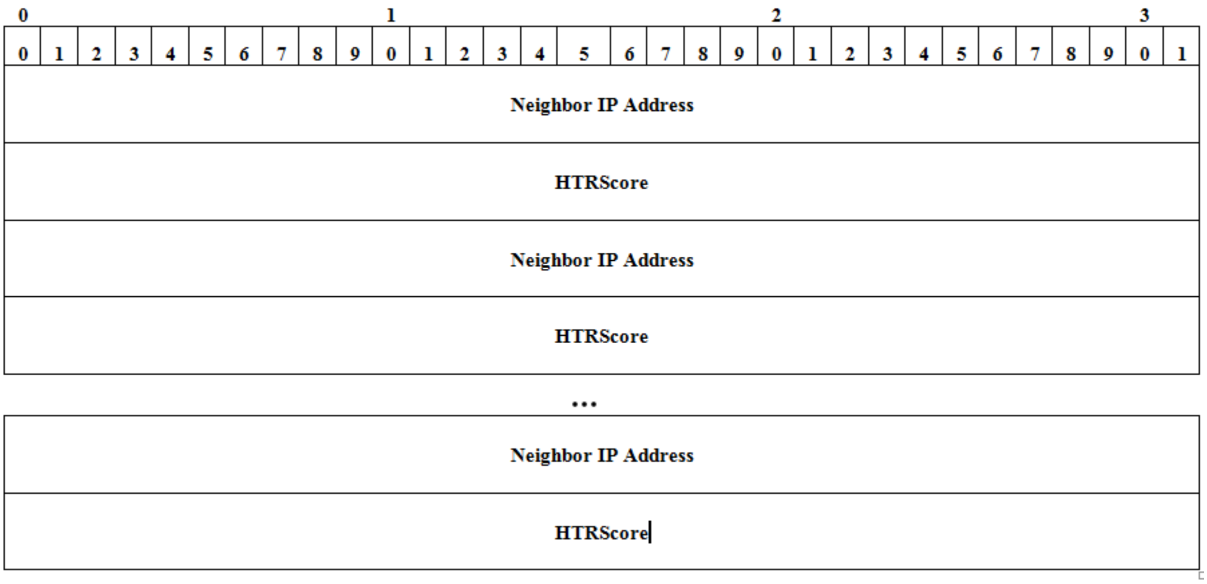


Figure 12: Data field format

1.1.1.1 *HELLO MESSAGE GENERATION*

Basically, a HELLO message is generated as follows:

1. The Validity field of the HTR message header is set to correspond to the value of the NEIGHB_HOLD_TIME interval, which is calculated as three times the HELLO refresh interval. The HELLO refresh interval is the amount of time between the sending of two HELLO messages, and has default value of 2 seconds for the OLSR and a value for HTR varying from 2.0 and 2.5 seconds. The HELLO refresh interval is evaluated in Chapter 5. The NEIGHB_HOLD_TIME is used to set the SYM_TIME, ASYM_TIME and TIME of the Link Neighbor Set Base record and represents the validity time for the information provided by the HELLO message.

2. The Message Type field of the HTR packet body is set to zero, which corresponds to the type HELLO.
3. Each Data field, which is part of the HELLO message and has a corresponding Link Code, must be computed as follows:
    a. For each entry of Link-Neighbor Set Base, where Time is greater than zero (i.e. not expired), the N_Addr field is advertised with:
        i. The Link Type (subfield of Link Code) is set according to the following:
            1. If SYM_Time is greater than zero (not expired) then Link Type is equal to the corresponding SYM_LINK value.
            2. If ASYM_Time is greater than 0 (not expired) and is not a SYM_LINK (see above), then the Link Type is equal to the corresponding ASYM_LINK value.
            3. If ASYM_Time has expired and SYM_Time has expired, then the Link Type is equal to the corresponding LOST_LINK value.
        ii. The Neighbor Type (subfield of Link Code) is set according to the following:
            1. If the address, corresponding to N_Addr, is included in the MPR Set Base, then Neighbor Type is equal to the MPR_NEIGH corresponding value.
            2. Otherwise, if the record in the Link-Neighbor Set base has:
                a. Status equal to SYM then Neighbor Type is equal to the SYM_NEIGH corresponding value.
                b. Otherwise, if it has status equal to ASYM then Neighbor Type is equal to the NOT_NEIGH corresponding value.
        iii. HTRScore is set according to the value computed by the algorithm in (1).

3.7.2.2 HELLO MESSAGE SENDING

The sending of HELLO messages is performed via broadcast to its 1-hop neighbors using all available interfaces, it is sent at every HELLO_INTERVAL, and must include the jitter considerations in order to decrease the probability of transmission collisions.

It is also important to emphasize that HELLO messages never are forwarded for more than one hop, since its TTL is 1.

3.7.2.3 HELLO MESSAGE PROCESSING

A node processes incoming HELLO messages for the purpose of conducting link sensing, neighbor discovery and MPR selector signaling, i.e. the HELLO message is the one to populate the Link-Neighbor Base, Two-Hop Neighbor Base and the MPR Selector Base. Upon receiving a HELLO message, these repositories are populated as follows:

1. If there exists no tuple of Link-Neighbor Set Base with N_Addr equal to the originator address (i.e. main address of the node which has emitted the HELLO message), then a new Link-Neighbor Set Base tuple is created with N_Addr equal to the originator source. This new tuple is created with L_Iface_ID the same index of the interface which received the HELLO message, with Status equal to ASYM, with HTR Score computed by the algorithm in (1) (see Section 1.4), with ASYM Time and Time fields equal to the Validity Time (i.e. the Validity Time field of the message header), which is equal to the NEIGHB_HOLD_TIME for the HELLO message.
2. For each Link Message contained in the HELLO message:
    a. If the node finds its own address among the addresses listed in the Link Message and there exists a tuple of Link Neighbor Set Base with N_Addr equal to the originator address and with L_Iface_ID equal to the receiving interface index, then the tuple is modified as follows:
        i. If the Link Type of the link message is equal to SYM_LINK or ASYM_LINK, then the Status of the tuple is equal to SYM, and the HTRScore is computed using the algorithm in (1), and the SYM TIME and TIME are equal to the Validity Time (NEIGHB_HOLD_TIME for the HELLO message).
    b. If the originator address is a symmetric 1-hop neighbor of the node, then:
        i. For each Neighbor IP Address contained in the Link Message, where the Neighbor Type is either MPR_NEIGH or SYM_NEIGH:
            1. If the Neighbor IP Address is equal to the receiving node address, then:

a. If the Neighbor Type is equal to MPR_NEIGH then
   i. If there exists no MPR selector tuple of MPR Selector Base with N_Addr equal to the originator address, then create a new tuple with N_Addr equal to the originator address and time equal to NEIGHB_HOLD_TIME.
   ii. Otherwise, update the Time field of the MPR selector tuple and set it to NEIGHB_HOLD_TIME.
b. If the Neigh Type is equal to SYM_NEIGH, then continue to the next Neighbor_IP_Address.

2. Otherwise, if there is no tuple in the Link Neighbor Base set where N_Addr is equal to the Neighbor IP Address of the Link Message
   a. If there is no tuple in the Two-Hop Neighbor Set Base where 2_Hop_N_Addr is equal to the Neighbor IP Address and the N_Addr is equal to the originator address, then create a new 2-hop tuple in the Two-Hop Neighbor Set Base with N_Addr equal to the originator address, and with HTRScore equal to the HTRScore provided by the correspondent Data field, with 2_Hop_N_Addr equal to the Neighbor IP Address and Time equal to the Validity Time (NEIGHB_HOLD_TIME for the HELLO message).
   b. Otherwise, update the HTRScore of the 2-hop neighbor tuple with the score provided by the corresponding Data field, and also, update the Time field of the 2-hop neighbor tuple to be equal to the Validity Time (NEIGHB_HOLD_TIME for the HELLO message).

ii. For each Neighbor IP Address contained in the Link Message, where the Neigh Type is equal to NOT_NEIGH and there exists a 2-hop neighbor tuple with the N_Addr equal to the originator address and the 2_Hop_N_Addr field equal to the Neighbor_IP_Address, then remove this tuple.

3. Compute MPR and populate MPR Set Base using the mechanism described in Section 3.8

3.7.3 TC MESSAGE

TC messages are the mechanism used to advertise topology information through the network using the optimized flooding mechanism through MPRs. The advertised topology information corresponds to the MPR-selector set that is disseminated to the entire network in order to provide sufficient information to enable routing. [71] gives a proof that the MPR flooding mechanism reaches all destinations in the network graph.

3.7.3.1 TC FORMAT

The format of the TC message is as follows (Figure 13):

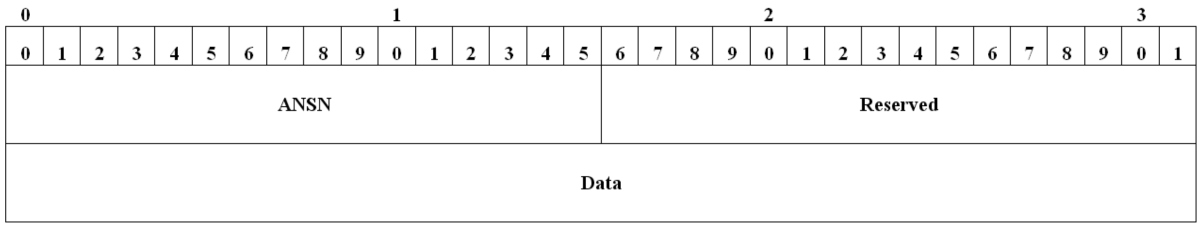


Figure 13: TC Message Format

Advertised Neighbor Sequence Number (ANSN): similar to OLSR, this field is associated with the advertised neighbor set (i.e. MPR selector set). Every time a node detects a change in its advertised neighbor set, it increments this sequence number (16 bits). This number is sent in this ANSN field of the TC message to keep track of the most recent information. When a node receives a TC message, it can decide on the basis of this ANSN, whether or not the received information about the advertised neighbors of the originator node is more recent than what it already has.

Reserved: a field reserved for future use (16 bits). It is filled with value integer "0".

Data: this field uses the same structure as the Data field used by the HELLO message and represents a variable field composed by the MPR selectors that includes the following information for each of these:

Destination IP Address: represents the address of the MPR selector (32 bits for IPv4 and 128 bits for IPv6).

HTRScore: represents the energy awareness metric used for path computation (32 bits). This metric represents the cost to transmit from the MPR node to its MPR selector with address equal to the “Destination IP Address” field.

3.7.3.2 TC MESSAGE GENERATION

In order to build the topology information base, each node selected as MPR broadcasts TC messages containing the MPR selector set. A node retransmits a broadcasted TC message only if it has received its first copy from a node for which it is an MPR.TC messages are sent and forwarded on a regular interval, the TC_INTERVAL, which is defined as 5 seconds for HTR and as well for the OLSR.

The MPR functionality introduces two optimizations to TC messaging that greatly reduce the overall network overhead:

Size optimization:

The size of TC messages is reduced, since a node only declares its MPR selectors through TC messages. Thus, the node does not have to send its whole repository sets.

Sender optimization:

A node that has no MPR selectors to declare usually does not transmit TC messages in order to save network resources. The exception here is nodes that have just lost their MPR selectors. These nodes generate empty TC messages for a given interval time equal to the Validity time of its previously emitted TC message, in order to invalidate the previous TC messages.

Basically, a TC message is generated as follows:

1. The Validity field of the HTR message header is set to correspond to the value of the TOP_HOLD_TIME interval, which is calculated as three times the TC_INTERVAL. The TOP_HOLD_TIME is used to set the TIME field of the Topology Control Base record and represents the validity time for the information provided by the TC message.
2. The Message Type field of the HTR packet body is set to 1, which corresponds to the type TC.
3. The ANSN field is only incremented when a change to the MPR selector set is detected; otherwise the ANSN must be set to the last one utilized.
4. Each Data field is computed as follows:
    a. For each entry of the MPR Selector Base, where Time is greater than 0 (i.e. not expired), the N_Addr field is advertised with the corresponding HTRScore computed using the algorithm in (1).
    b.

3.7.3.3 *TC MESSAGE FORWARDING*

TC messages are broadcast and retransmitted by the MPRs in order to diffuse the messages into the entire network. TC messages can be forwarded according to the "default forwarding algorithm".

3.7.3.4 *TC MESSAGE PROCESSING*

Upon receiving a TC message, the TC repository is updated as follows:

1. If the sender (not the originator), that is, the node broadcasting or forwarding the message, is not in the symmetric 1-hop neighborhood, the message is discarded.
2. If there is some tuple in the Topology Control Base set where T_Last_Addr is equal to the originator address (i.e. originator address of the packet body header)and the Seq (i.e. the field in the Topology Control Base record) is greater than the ANSN, then further processing of this TC message is not performed and it is discarded.
3. All tuples in the Topology Control Base set where T_Last_Addris equal to the originator address and Seq is less than the message ANSN, are removed.
4. For each Destination IP Address received in the TC message:
    a. If there exists some tuple in the topology set where T_Dest_addr is equal to the Destination IP Address, and T_Last_addr is equal to the originator address, then the holding time (Time field) must be set to the Validity Time (i.e. message header field), which is, for the TC message, equal to the TOP_HOLD_TIME, and the HTRScore field is set to the corresponding HTRScore for the Destination IP Address contained in the TC message.
    b. Otherwise, a new tuple is recorded in the Topology Control Base set where: T_Dest_addr is equal to the Destination IP Address, T_Last_addr is equal to originator address, HTRScore is equal to the correspondent HTRScore of the Destination IP Address, T_Seq is equal to ANSN, Time is equal to the Validity Time, which is equal to TOP_HOLD_TIME for the TC message.

3.8 FLOODING MECHANISM

As previously mentioned, HTR protocol employs an optimized flooding scheme to diffuse topology information throughout the network. The use of a flooding scheme without any optimization for MANET context causes an intense waste of network resources, which are limited by the wireless nature of the medium that limits the bandwidth capacity available in a frequency band. Thus, every protocol that is going to use wireless links has to keep its unnecessary traffic to a minimum. The simplest way of broadcasting a packet to all nodes in the network is by basic flooding or blind flooding, which allows each node to retransmit a packet to its neighbors only if it has not received the packet before. This rebroadcasting continues until all the nodes in the network received a copy of the packet. The main disadvantage of blind flooding is that it can trigger a large number of packets forwarded in MANETs which will eventually overwhelm the network [72]. Figure 14 illustrates the blind flooding scheme, where each node retransmits each packet received and results in overly redundant rebroadcasting (i.e. some nodes receive the same packet more than once), contention and collision (broadcast storm problem [73]). The use of an optimization flooding scheme is required to ensure better use of network resources.

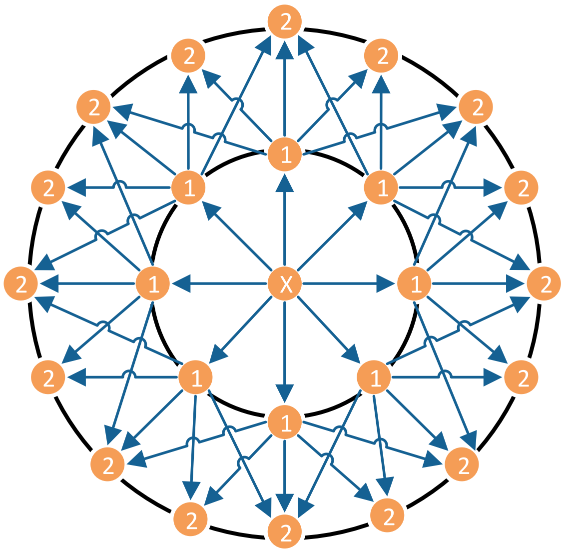


Figure 14: Regular Flooding

3.8.1 MULTIPOINT RELAYING

The Multipoint Relay (MPR) main goal is to reduce the flooding of broadcast packets in the network by minimizing redundant retransmissions locally. The Multipoint Relay (MPR) technique, a neighbor designated method that exhibits both efficiency and simplicity, uses a simple algorithm to calculate the forwarding nodes, which makes it easy to implement. Furthermore, it has been successfully employed in both sparse and dense networks. The concept of MPR was first introduced in the [32], which was a MAC layer protocol developed by the European Telecommunications Standards Institute (ETSI) to provide a substitute for wired LAN and was then successfully extended to MANET context.

Using the MPR technique, every node in the network selects a subset of its 1-hop neighbors, called multipoint relays (MPRs), as the forwarding node set to retransmit broadcast messages. It calculates its own set of MPRs as a subset of its symmetric 1-hop neighbors chosen so that all 2-hop neighbors can be covered through its MPR. The MPR flooding scheme is illustrated in Figure 15.

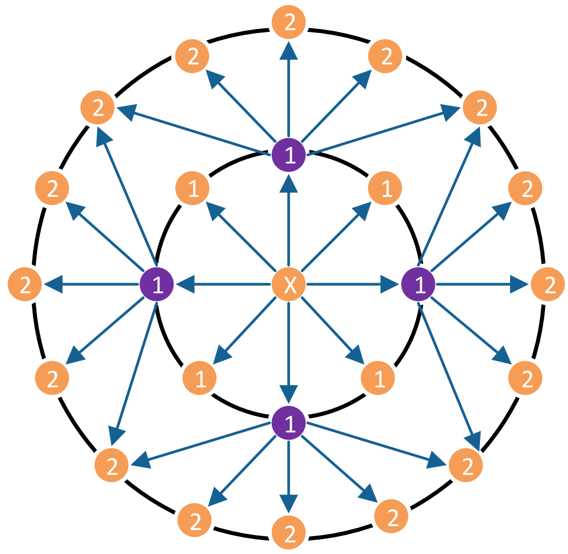


Figure 15: Multipoint Relay Flooding

Several algorithms have been specified to achieve, in an efficient manner, the MPR set. Finding the optimal MPR set has been proved to be a NP-complete problem in [74] and has been further studied and analyzed in, amongst others, [75]–[77].The HTR protocol uses a method for MPR selection based on HTRScore, where the energy consumption and link stability are considered to ensure better network efficiency. In general lines, the MPR proposal considers that each node participating in the selection of its MPR set may have distinct priorities (e.g. energy awareness) that could change during its lifetime. Therefore, it should use any technique that takes into account the impact of each metric in the selection phase. Using the scheme below, the protocol ensures that nodes could choose candidates (MPRs) with better benefits for the actual context, increasing the performance for the whole network.

3.8.2 SELECTION OF MPR SET

Initially, the terminology used is supplied to ensure an acquaintance with the process described:

N: A set containing all 1-hop neighbors.

N2: A set containing all 2-hop neighbors.

P: the set of all members of N2 that are not yet covered from any node contained in the MPR set.

C(Y): the number of members of P reached directly from Y (a 1-hop neighbor) in a symmetric way.

D(Y): the number of symmetric neighbors reached directly from Y, a 1-hop neighbor, excluding all other 1-hop neighbors and the node performing the computation.

To fulfill all these requirements, the method utilizes the HTRScore as the cost of the path between two nodes. The HTRScore takes into account the order of transmission; nevertheless, the cost of the path between nodes A and B could be different from the cost of the path from B to A.

The HTRScore is calculated by the local node for each symmetric neighbor and also is received from other nodes by the use of HELLO messages and also TC messages.

The method for selection of MPR proposed in this specification is the following:

1. Start with an MPR set empty.
2. Fill P with all members of N2.
   a. Add to the MPR set those nodes in N, which are the only nodes to cover a node in N2. For example, if node B in N2 can be reached only through a symmetric link to node A in N, then add node A to the MPR set.
   b. Remove the nodes from P that are now covered by a node in the MPR set.
3. While there are nodes in P:
   a. For each node in N, compute or use the HTRScore(Y) using actual weighting factors related with the node Y.
   b. Select as an MPR the node with lowest non-zero HTRScore among the nodes in N.
      i. For the case in which multiple nodes have the same HTRScore, calculate the C(Y) of these nodes and select the node with maximum C(Y).
      ii. If the C(Y) value is the same for all nodes, calculate the D(Y) and select the node with maximum D(Y).
      iii. If the D(Y) value is the same, select the node that has been chosen as MPR less times.
      iv. If still the values are the same, select the first node of the remaining set.
   c. Remove the nodes from P that are now covered by the selected node in the MPR set.

3.8.3 EXAMPLES

In order to explain more specifically the MPR selection process, this subsection presents two example scenarios: the first to illustrate a simple and symmetric scenario and the last to represent a more realistic scenario.

3.8.3.1 Scenario 1

The node A, its four 1-hop neighbors, and the eight 2-hop neighbors compose this scenario, which is illustrated in Figure 16.

To perform the MPR selection, node A use its knowledge about its neighbors to mount N, N2 and MPR set. Following the first steps to select MPRs, the node A has the following sets:

N = {B, C, D, E},

N2 = {F, G, H, I, J, L, M, N},

MPR = { },

P = {F, G, H, I, J, L, M, N }.

Following the step 2, the node A observes that pairs of 2-hop neighbors are only covered by a unique 1-hop node at a time (F and G are reached by B, H and I are reached by C, J and L are reached by D, and M and N are reached by E). This way, according to the step, the nodes that only are covered by a unique 1-hop node are removed from the P set and these 1-hop neighbors are added to the MPR set. As result, the MPR and P sets are:

MPR = {B, C, D, E},

P = { }.

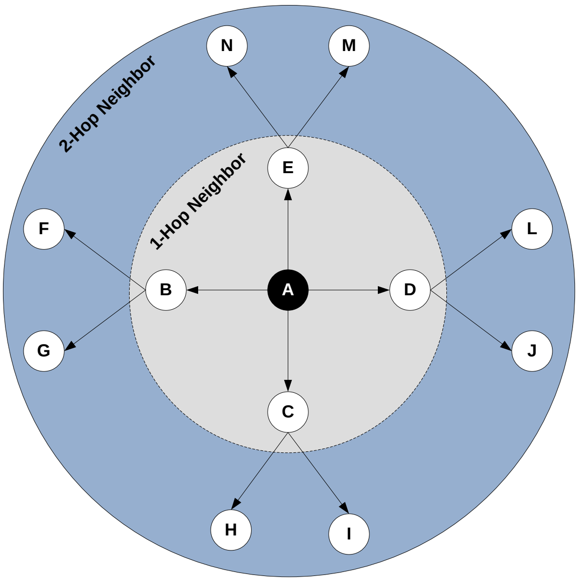


Figure 16: MPR selection for Scenario 1

It is important emphasize that nodes added in the MPR set cannot be added again.

3.8.3.2 SCENARIO 2

Scenario 2 reflects a more realistic scenario and is capable of explaining almost all steps of the MPR selection process. Figure 17 shows that node A has nine 1-hop neighbors and each of these have at least two 2-hop neighbors covered. Unlike the first example, this scenario presents 2-hop neighbors being covered by two or three 1-hop neighbors at a time, which implies more possibilities to choose as MPR. In addition, this scenario also presents the calculated HTRScore (the cost of edges) between each pair of nodes.

Node A knowledge about its neighbors and MPR selection elements are represented in the following way:

N = {B, F, H, C, G, D, I, E, J},

N2 = {K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z},

MPR = *{ }*,

P = {K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z}.

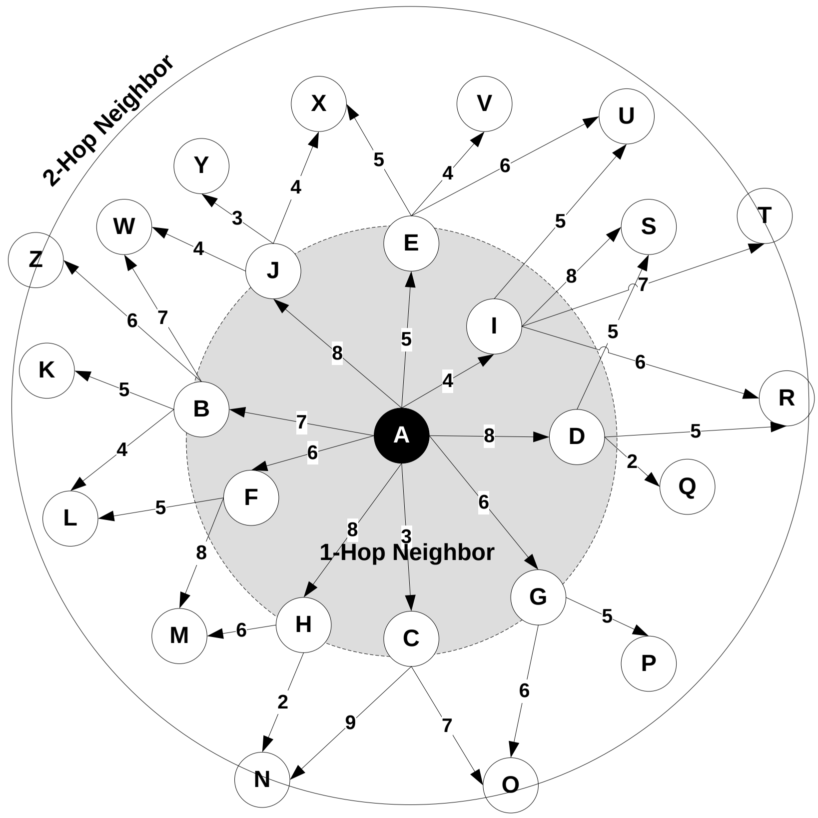


Figure 17: MPR selection for a realistic scenario (Scenario 2)

According to scenario 2, node A observes that some 2-hop neighbors are only covered by some specific 1-hop neighbors. This way, according to the scenario, the following operations happen:

1. Node Z is only covered by node B. For this reason, node Z is removed from the P set and node B is added to the MPR set.
2. Node K is only covered by node B. For this reason, node Z is removed from the P set. Node B is not added in the MPR set because it was already added previously.
3. Node P is only covered by node G. For this reason, node P is removed from the P set and node G is added to MPR set.
4. Node Q is only covered by node D. For this reason, node Q is removed from the P set and node D is added to the MPR set.
5. Node T is only covered by node I. For this reason, node T is removed from the P set and node I is added to the MPR set.
6. Node V is only covered by node E. For this reason, node V is removed from the P set and node E is added to the MPR set.
7. Node Y is only covered by node J. For this reason, node Y is removed from the P set and node J is added to the MPR set.

As a preliminary result of this step, the sets of node A are:

MPR = {B, G, D, I, E, J},

P = {K, L, M, N, O, R, S, U, W, X}

Figure 18 illustrates the partial selection of MPR, where the blue nodes are added in the MPR set and the yellow nodes are removed from the P set.

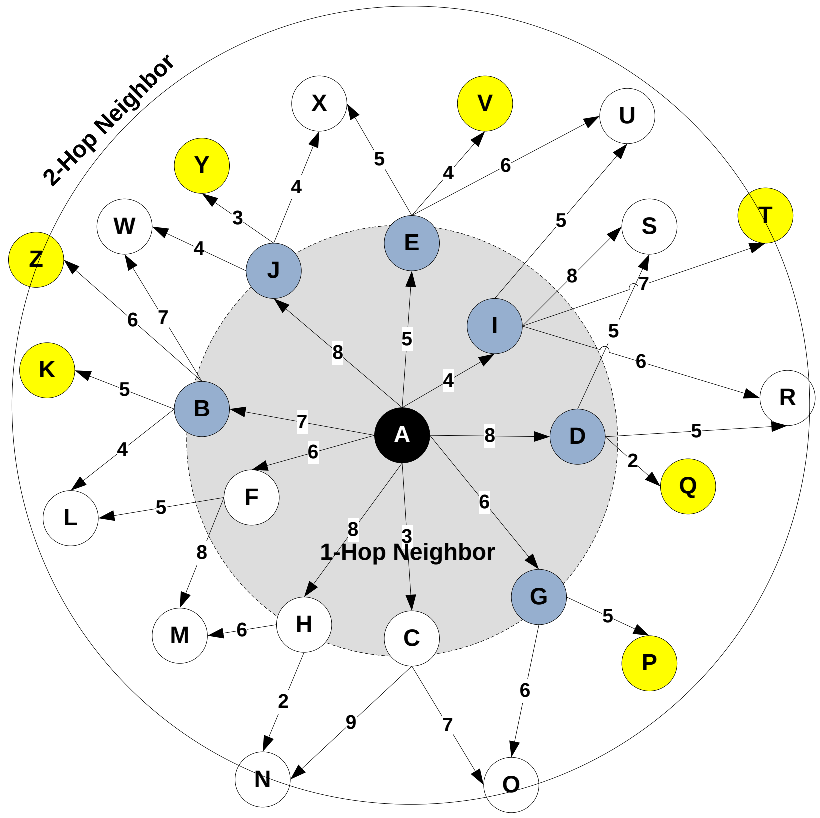


Figure 18: Partial result of MPR selection

However, there are still nodes in the P set. This way, the next step is started. Step 3 employs the HTRScore to select MPRs and still presents some mechanism to deal with conflicts. This way, node A will use the HTRScore of each path as follows:

1. Node L has two possible values for HTRScore (two options for the path). The path passing by node B has HTRScore equal to 11 (7 from A to B plus 4 from B to L) and the path via node F has HTRScore equal to 11 (6 from A to F plus 5 from F to L). In this case, a draw occurs.
    a. According to the first criteria of conflict, node A calculates the C(Y) from B and F, remembering that C(Y) represents the nodes covered that still are in P. So, the C(B) is equal to 2 (W and L), since the nodes Z and K were previously removed from P. The C(F) is also 2 (L and M). Thus, once again a draw occurs.
    b. According to the second criteria of conflict, node A calculates the D(Y) from B and F, remembering that D(Y) represents the nodes covered by a given node. Thus, the D(B) is equal to 4(W, Z, K, and L) while the D(F) is equal to 2 (L and M). In this case, node B is chosen as MPR and node L is removed from P set.
2. Node M has two possible values for HTRScore. The value via node F is equal to 14

(6 from A to F plus 8 from F to M) and via node H the value of HTRScore is equal to 14 (8 from A to H plus 6 from H to M). In this case, a draw occurs.

   a. Using the first criteria of conflict, the C(Y) from F is equal to 1 (M), since node L was already removed from P after node B has been chosen as its MPR. The C(Y) from H is equal to 2 (M and N nodes). In this case, node B is chosen as MPR and node L is removed from the P set.

3. Node N also has two possible values for HTRScore. The value via node H is equal to 10 (8 from A to H plus 2 from H to N) and via node C the value of HTRScore is equal to 12 (3 from A to C plus 9 from C to N). In this case, node H is chosen as MPR, because its value is lower than the one from node C. The node N is removed from the P set.
4. Node O also has two possible values for HTRScore. The value via node C is equal to 10 (3 from A to C plus 7 from C to O) and via node G the value of HTRScore is equal to 12 (6 from A to G plus 6 from G to O). As result, node C is added in the MPR set and node O is removed from the P set.
5. Node R also has two possible values for HTRScore. The value via node D is equal to 13 (8 from A to D plus 5 from D to R) and via node I the value of HTRScore is equal to 11 (4 from A to I plus 7 from I to R). As result, node I is added in the MPR set and node R is removed from the P set.
6. Node S also has two possible values for HTRScore. The value via node D is equal to 13 (8 from A to D plus 5 from D to S) and via node I the value of HTRScore is equal to 12 (4 from A to I plus 8 from I to S). As result, node I is added in the MPR set and node S is removed from the P set.
7. Node U also has two possible values for HTRScore. The value via node I is equal to 9 (4 from A to I plus 5 from I to U) and via node E the value of HTRScore is equal to 11 (5 from A to E plus 6 from E to U). As result, node I is added in the MPR set and node U is removed from the P set.
8. Node X also has two possible values for HTRScore. The value via node E is equal to 10 (5 from A to E plus 5 from E to X) and via node J the value of HTRScore is equal to 12 (8 from A to J plus 8 from J to X). As result, node E is added in the MPR set and node X is removed from the P set.
9. Node W also has two possible values for HTRScore. The value via node J is equal to 12 (8 from A to J plus 4 from J to W) and via node B the value of HTRScore is equal to 14 (7 from A to B plus 7 from B to W). As result, node J is added in the MPR set and node W is removed from the P set.

As a final result, node A has its MPR set composed of the following nodes:

MPR = {B, H, C, G, D, I, E, J}

It is important emphasize that the MPR selection mechanism does not insist that all 1-hop neighbor nodes be chosen as MPR nodes, as illustrated by this example. The MPR set populates the MPR Set Base where, for each MPR contained in the computed set, one new tuple is created with N_Addr equal to the address of the MPR node and the Time field set as the NEIGHB_HOLD_TIME. The MPR computation is computed each time the HELLO message is processed.

3.9 PATH COMPUTATION

By using a proactive approach, HTR ensures that a path is computed as early as possible but, with the conventional path computations that focus only on hops to reach a destination, the energy of the nodes along those paths is rapidly burned, and eventually disconnects nodes, partitioning the network.

One of HTR goals is to maintain network connectivity and survivability. In other words, the protocol should ensure that connectivity is maintained for as long as possible. This is done by applying a modified version of the Dijkstra Algorithm [78], called Multipath Dijkstra (see next section), which uses HTRScore as the path cost function and also computes multiples paths to reach destinations. The concept of energy awareness was already considered in [79]; however, the HTRScore, as previously mentioned in Section 3.4, also considers the link stability that is not covered by [79].

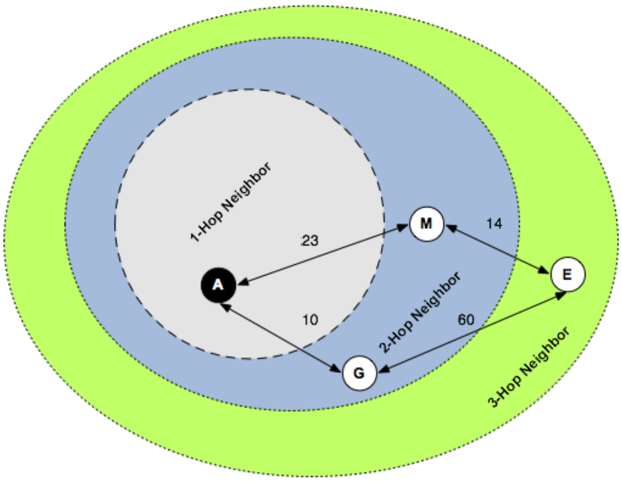
Dijkstra's algorithm is a graph search algorithm that solves the single-source shortest path problem for a graph with nonnegative edge path costs, producing a shortest path tree. Considering the initial node and the distance for node Y as the distance (cost) from the initial node to node Y, Dijkstra's algorithm will assign some initial distance values and will try to improve them step-by-step.

1. Assign to every node a distance value. Set it to zero for the initial node and to infinity for all other nodes.
2. Mark all nodes as unvisited. Set initial node as current.
3. For the current node, consider all its unvisited neighbors and calculate their distance (from the initial node). For example, if current node A has distance of 6, and an edge connecting it with another node B is 2, the distance to B through A will be 8. If this distance is less than the previously recorded distance (infinity in the beginning, zero for the initial node), overwrite the distance.
4. When concluding the calculations for all neighbors of the current node, mark it as visited. A visited node will not be checked ever again; its distance recorded now is

final and minimal.

5. Set the unvisited node with the smallest distance (from the initial node) as the next "current node" and continue from step 3.

For the result of the process above, all minimum distances (i.e. minimum cost) to reach destinations are obtained. Each path computed is used to maintain the routing table of the node; however, it is only necessary for the next hop to be passed to the route table for each path observed, i.e. only the closest intermediate node contained in the path is applied at the routing table, since the same process is applied over each node of the domain, and the path is constructed in a distributed manner as illustrated in the Figure 19.



Route Table - A	
Destination	Gateway
E	M
M	
G	

Route Table -M	
Destination	Gateway
A	
E	
G	A

Route Table - E	
Destination	Gateway
A	M
M	
G	

Route Table - G	
Destination	Gateway
A	
E	
M	A

Figure 19: Route Calculation and Route Table samples

3.10 MULTIPATH DIJKSTRA

Unlike the single path strategy, with a multipath approach different paths are computed between source and destination. Multipath routing could offer several benefits [80] such as load balancing, fault tolerance, routing loop prevention, higher aggregate bandwidth, energy-conservation, lower end-to-end delay, security, and congestion avoidance [30], [81].

During the HTR evaluation, we noticed the occurrence of routing loops, which were significantly decreasing the performance of the protocol. The presence of these loops was associated with the HTRScore computation, which, in some cases, was producing divergent views of the network. Considering that backup routes could be used to prevent or effectively decrease the occurrence of routing loops, a new process for path computing was developed. This new process uses a multipath routing algorithm based on the algorithm introduced in [56], the *MultipathDijkstra*, to mitigate the routing loops. Our modified version of the *MultipathDijkstra* is shown in Figure 20.

```
MultiPathDijkstra (s, d, G, N)
    HTRScore_1 ← HTRScore
    G_1 ← G
    for i ← 1 to N do
        SourceTree_i ← Dijkstra (G_i, s)
        P_i ← GetPath (SourceTree_i, d)
        if e or Reverse (e) is in P_i then
            HTRScore_{i-1} ← f_p (HTRScore_i (e))
        else if Head (e) is in P_i then
            HTRScore_{i-1} (e) ← f_e (HTRScore_i (e))
        else
            HTRScore_{i-1} (e) ← HTRScore_i (e)
        end if
    end for
```

Figure 20: HTR Multipath Dijkstra

The general principle of this algorithm occurs at step i, which looks for the shortest path Pi to the destination d. Then the edges of Pi or pointing to Pi have their cost increased in order to prevent the next step from taking this same path again. The fp function is used to increase the cost of arcs belonging to the previous path P_i. This encourages future paths to use different arcs but not different vertices. f_e is used to increase the cost of the arcs that would lead to the vertices of the previous path, P_i. The f_p and f_e functions are used to get link-disjoint paths or node-disjoint routes as necessary. The *Dijkstra* (G_i, s) is the standard Dijkstra's algorithm which provides the source tree of shortest paths from vertex s in graph G; *GetPath(ST,d)* is the function that extracts the shortest path to d from the source tree ST; the function *Reverse(e)* gives the opposite edge of e; *Head(e)* provides the vertex edge e points.

To avoid the creation of similar paths, the f_p and f_e functions consider vertices and edges used in previous paths. In contrast to the approach presented in [19], which updates the scores by multiplying them by a constant, our proposal multiplies them (the HTRScores) by the number of times each vertex/edge was used.

To prevent routing loops, the approach adds a header to the packet whereupon every node of the path records its identification, creating a list of nodes which have already processed a packet. When received, the node inspects the header of the packet and searches the routing table for the next node to which it is possible to forward the packet, considering the backup paths defined, but excluding the nodes already on the list. If there is no alternative path available, the packet is discarded to avoid unnecessary use of resources. The header is removed when the packet is received by the destination node. The multipath routing algorithm computes a default value of ten distinct paths to any given destination. This value may not be the optimal one; however, it drastically decreases the number of routing loops to a minimum. The analysis of the multipath HTR routing table computation process is presented in Chapter 5.

3.11 CHAPTER SUMMARY

In this chapter, the concepts and functionalities of the HTR protocol were introduced. Initially, an initial overview of the protocol, which includes the main characteristics and features of this one, and the terminology were provided. Secondly, the process for address allocation and maintenance, named DNCP, was commented. The DNCP employs a distributed generation function that allocates a unique identifier for each node. Third, the HTRScore cost metric, which uses factors such as the awareness of link conditions and power efficiency, was defined and is used to perform path computation and MPR set computation. Fourth, the HTR repositories, which represent the network topology and are updated regularly via control messages, were presented. Next, HTR packet was introduced and the format, processing and forwarding process for this one were provided. Also, the considerations for jitter, which is a random small time interval for which message has to be cached before it is transmitted to avoid collisions, were discussed. Furthermore, the HTR control messages, the HELLO and TC messages, were detailed and the format, generation, and processing of these ones were provided. In addition, the HTR MPR optimization, which is used for efficient message flooding, was defined and uses the HTRScore for MPR set computation. Finally, the path computation process and the HTR Multipath Dijkstra algorithm, proposed in this work, were defined. The HTR multipath approach computes different paths between source and destination.

CHAPTER 4 HTR IN NS-3

Computer simulating is one effective method [82] for researching difficult systems since computer models are easier, lower cost and more convenient to investigate than real experiments. In the field of communication systems engineering, the network simulation is are widely used for the development of new communication architectures and network protocols. For example, in order to investigate the characteristics of a new routing protocol (i.e. routing behavior), different topologies, which are merely sets of simulation parameters, can be easily utilized with a network simulator. Most of these are based on the paradigm of discrete event-based trigger events (DES) [83]. Here, the simulator maintains an event queue sorted by the scheduled event execution time and the simulation is performed successively, processing the events contained in this queue. A simulation event is triggered, for instance, by the sending of a packet from one node to another. The first approaches where DES was applied to the simulation of computer networks were published about two decades ago and ns-2 is a direct successor of those early efforts [84].

To implement and evaluate the performance of the proposal, the HTR protocol was implemented using a network simulator. In this chapter, initially an introduction and evaluation of the network simulators, the ones that have gained attention in the research field, are provided in order to discuss the reason for the use of the ns-3 network simulator during this work. Next, the details of the implementation of the HTR protocol in the ns-3 simulator are provided and the extensions required for the operation of the protocol are also shown.

4.1 NETWORK SIMULATORS

During the elaboration of this work, we tried to evaluate some of the open source widespread network simulators that are in use today using [84], [85] as reference. This evaluation (see Table 7) had the objective of identifying the network simulator to implement the work discussed herein. The metrics utilized are: *Core Efficiency* and *Heterogeneous Technologies Support*. The first one uses the experimental analysis of [84], [85], which evaluates the scalability, computation time, and memory usage of those network simulators. The second metric checks whether each heterogeneous technology required by this work (i.e. Wi-Fi, WiMAX, and LTE) is supported by the network simulator. Each metric was ranked from lowest to highest score, respectively, as follows: Low / Insufficient (+), Medium (++) and High (+++). For the second criterion (i.e. Heterogeneous Technologies Support), the "+" indicates whether the simulator supports the technology and the absence attests to the opposite. Finally, the overall score for each network simulator was computed, wherein each

"+" represents one point, and we concluded that the ns-3 is the network simulator to be used for this work. Next, an introduction for each network simulator compared is given.

Table 7: Comparison of network simulator

Metric	ns-2	ns-3	OMNeT++	JiST	SimPy
Core Efficiency					
Scalability	+	+++	+++	+++	+
Performance	+	++	++	+++	+
Memory usage	+	+++	++	+	+
Heterogeneous Technologies Support					
IEEE 802.11	+	+	+		
WiMAX	+	+	+		
LTE	+	+	+		
Total	**6**	**11**	**10**	**7**	**3**

4.1.1 NS-2

The ns-2 [86], initially released in 1996, has become the standard for network simulation, which can be attributed to the fact that numerous models such as protocol models and traffic generators are publicly available, thus eliminating the need to implement them by hand. However, a major deficiency of ns-2 is its limited scalability in terms of memory usage and simulation run-time [23], [87], an especially serious problem for new research that requires the simulation of very large networks, potentially with hundreds of thousands of nodes. In order to face those challenges, some enhancements of ns-2 have been proposed such as the incorporation of parallelization, although ns-2 is currently being replaced by its successor, the ns-3 that has, as one of its goals, the improvement of simulation performance.

Simulations for ns-2 are composed of C++ [88] code, used mainly to model the behavior of simulation components, and oTcl [89] scripts with simulation parameters, for instance the network topology. The use of oTcl was originally made back in 1996 to avoid unnecessary C++ recompilations when changes are made to the simulation set-up.

4.1.2 OMNeT++

The "Objective Modular Network Testbed in C++" (OMNeT++) [90] simulator is an extensible, modular, object-oriented, component-based C++ simulation library and framework. In contrast to ns-2 and ns-3, it is not a network simulator by definition, but a general purpose discrete event simulation framework. It can be used for simulating any general network that includes wired and wireless communication networks, on-chip networks, sensor networks, ad hoc networks, and photonic networks, etc. These are provided by model frameworks, developed as independent projects. The simulator offers an Eclipse-based IDE, a graphical runtime environment and a host of other tools.

The OMNeT++ architecture consists of simple modules and compound modules. Simple modules are atomic elements (i.e. they cannot be divided any further) in the module hierarchy and can be connected with each other via *gates* to form compound modules. These compound modules and the setup of the entire network topology takes place in a language known as Network Descriptor (NED) that enables the assignment of network parameters statically (i.e. before the execution) or dynamically (i.e. during run-time).

4.1.3 JiST

JiST [91] is a high-performance discrete event simulation engine that runs on a Java virtual machine [92]. Simulation in JiST is made up of entities that actually represent the network elements that use class methods to invoke simulation events. Embedding the simulation semantics within the Java language enables JiST to inherit all the properties and libraries of Java including the existing compilers. JiST benefits from the automatic garbage collection, type-safety, reflection and many other properties of that language. Unfortunately, the official development of JiST has stalled and it is no longer maintained.

4.1.4 SimPy

SimPy [93] is a process-oriented discrete event simulator written using the Python language. In this simulator, the basic simulation entities are processes that are executed in parallel and may exchange Python objects among one another. Most processes include an infinite loop machine wherein its internal behavior is defined. Unlike the other simulators, no available network models are available for it and SimPy can be merely be classified as a bare simulation API written in Python [84].

4.1.5 ns-3

Like its predecessor ns-2, ns-3 relies on C++ for the implementation of the simulation models. Nevertheless, ns-3 no longer uses oTcl scripts to control the simulation. Instead, ns-3 simulations can be implemented in pure C++ or optionally using Python [94] as well. Moreover, the simulator integrates architectural concepts and code from GTNetS [95], a simulator with good scalability characteristics. These design decisions were made at the expense of compatibility, thus forcing ns-2 models to be ported to ns-3 in a manual way. Besides performance improvements, the feature set of the simulator has also been extended.

The ns-3 allows researchers to study network protocols and large-scale systems in a controlled environment. It is a new simulator aimed at eventually replacing the ns-2 simulator and there is no backward compatibility between them [96]. Some models from ns-2 have already been ported to the new simulator. It works under GNU / Linux [97], although there are various methods for using it under Microsoft Windows [98]–[100].

The Network Simulator 3, also known as the ns-3, was started in 2006 and is a discrete event network simulator written in C++. It is used primarily for research and educational applications aligned with modern network research.

The ns-3 is free software and uses the GNU GPLv2 [101] as license, which guarantees end users (individuals, organizations and companies) the freedom to use, study, share (copy), and modify the software. The ns-3 project encourages community contributions. Table 8 represents the main contributions of last releases [102]:

Table 8: ns-3 Project Progress since 2008

Release	**Contributions**
ns-3.1: June 2008	CSMA Point-to-Point **IEEE 802.11** TCP **UDP** **IPv4** **OLSR** **Mobility Models**
ns-3.4: April 2009	Tap Device **WiFi Models** ICMP
ns-3.5: July 2009	802.11e MAC EDCA 802.11b PHY
ns-3.6: October 2009	802.11s mesh IPv6
ns-3.7: January 2010	802.11p PHY AODV NetAnim
ns-3.8: April 2010	**WiMAX**
ns-3.9: August 2010	**Energy Model**
ns-3.10: January 2011	**WiFi Energy Model** **PyViz visualizer** **802.11g PHY**
ns-3.14.1: June 2012	**The LTE code from the LENA project merged** Dynamic Source Routing protocol
ns-3.15: August 2012	Maintenance release: *ns-3* models have been fixed

From Table 8, one can see that the ns-3 project is very flexible, provides actual protocol stacks and gives us an opportunity to work with the newest technologies in virtual environments. To accomplish this work, version 3.15 was used and modified. These modifications are explained in the chapter. The contributions that were added to the ns-3 project over the past several years and have been essential for the progress of the work described within this dissertation are highlighted in bold (see Table 8).

For more details about the operations of the core components of the ns-3 software, one can refer to Appendix A. The next sections, outline all of the contributions added which are necessary for the implementation of the simulated version the Heterogeneous Routing Protocol (HTR) described in Chapter 3.

4.2 HTR

This section describes the core implementation parts of the HTR routing protocol for the ns-3 simulator. The ns-3 is intended to support traditional approaches to routing protocols and facilitate research into unorthodox routing techniques. The architecture is designed to support different routing approaches, including policy-based routing implementation, proactive and on-demand routing protocols, and so on. The following section describes the HTR ns-3 routing model design.

4.2.1 HTR ROUTING MODEL DESIGN

The key objects are *Ipv4L3Protocol*, *Ipv4RoutingProtocol*(s) and *Ipv4Route*(s). The first one implements the IPv4 protocol. The second (see Appendix A) is the class which delegates the routing and forwarding of packets to *Ipv4L3Protocol*. *Ipv4L3Protocol* must have at least one *Ipv4RoutingProtocol* (e.g. *HTRRoutingProtocol*) attached to it during simulation setup time (i.e. simulation configuration prior to its execution). The last class represents the IPv4 route object.

The abstract base class *Ipv4RoutingProtocol* declares a minimal interface, consisting of two methods (see Appendix A): *RouteOutput* and *RouteInput*. For packets traveling outbound from a host, the transport protocol will query *Ipv4RoutingProtocol*, which is the *HTRRoutingProtocol* when using the HTR protocol, and will request a route via *RouteOutput* method. The HTR *Route Output* method lookups for a route, which is computed by *MultipathDijkstra,* that delivers the packet to the destination. The route contains the next reachable node (intermediate node) to forward the packet until reaching the destination, and for preventing routing loops, it adds an additional header to the packet to be forwarded (transmitted), a header which first contains its own identification (i.e. address), whereupon the intermediate node contained in the path records its identification, creating a list of nodes that have already processed a packet. The route returned by this method is represented by the *Ipv4Route* class. *Ipv4Route* objects contain the source, destination, gateway addresses and also the output interface. The gateway address represents the next hop to route the packet (if necessary).

For packets received inbound for forwarding or delivery, the *RouteInput* method is used. This one passes the packet ownership to the *Ipv4RoutingProtocol* object that is represented by the *HTRRoutingProtocol*. There are four possible actions associated with this call: *LocalDeliver*, *IpForward*, *IpMulticastForward* and *DropPacket*. The HTR *RouteInput* method works as follows:

Initially, the *RouteInput* method inspects the additional header, added during the *RouteOutput* method, to detect the list of nodes that have already processed it and also marks its identification (i.e. address) on the header. Next, if the packet must be forwarded elsewhere (*IpForward* or *IpMulticastForward*), it searches the routing table for the next node to which it is possible to forward the packet, considering the backup paths defined during the multipath path computation, but excluding the nodes that have already processed the packet. If there is no alternative path available, the packet is discarded (*DropPacket*), to avoid unnecessary use of resources. The header is removed when the packet is received by the destination node (*LocalDeliver*).

The Figure 21 illustrates how multiple routing protocols derive from the *Ipv4RoutingProtocol* base class. The HTR routing protocol is represented by the *HTRRoutingProtocol* class and derives from it. To implement a routing protocol using ns-3, it is necessary to implement the *RouteInput* and *RouteOutput* methods.

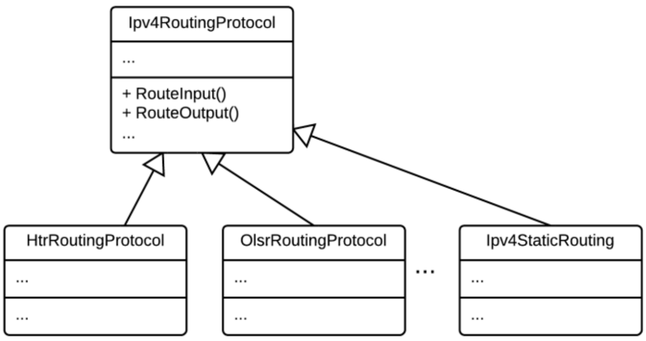


Figure 21: ns-3 IPv4 routing derived classes

4.2.2 HTR DESIGN

The HTR protocol, designed on ns-3, was divided into three core parts: *HTR State*, *HTR Routing* and *HTR Header*. The first one manipulates the HTR repositories and provides functions to add, retrieve, modify and destroy records. The second is the component for the sending and processing of control messages, routing path computation, MPR set computation, HTRScore computation, and for maintaining the consistency of its repositories. The third is the one for construction, manipulation, serialization and deserialization of control messages.

The *HTR State* contains the following tuple entries: *LinkNeighborTuples* and *NeighborTuple* (both representing 1-hop neighbors), *TwoHopNeighborTuple* (2-hop neighbors), *TopologyControlTuple* (Topology Control Base record), *DuplicateTuple* (marks the control messages already processed to avoid unnecessary processing of repeated messages), *MPRTuple* (each neighbor chosen as MPR during the MPR computation) and the *MPRSelectorTuple* (each neighbor that chosen the node as MPR). Figure 22 shows the UML class diagram that models the *HTR State* and includes the set of attributes for each tuple entry contained in it.

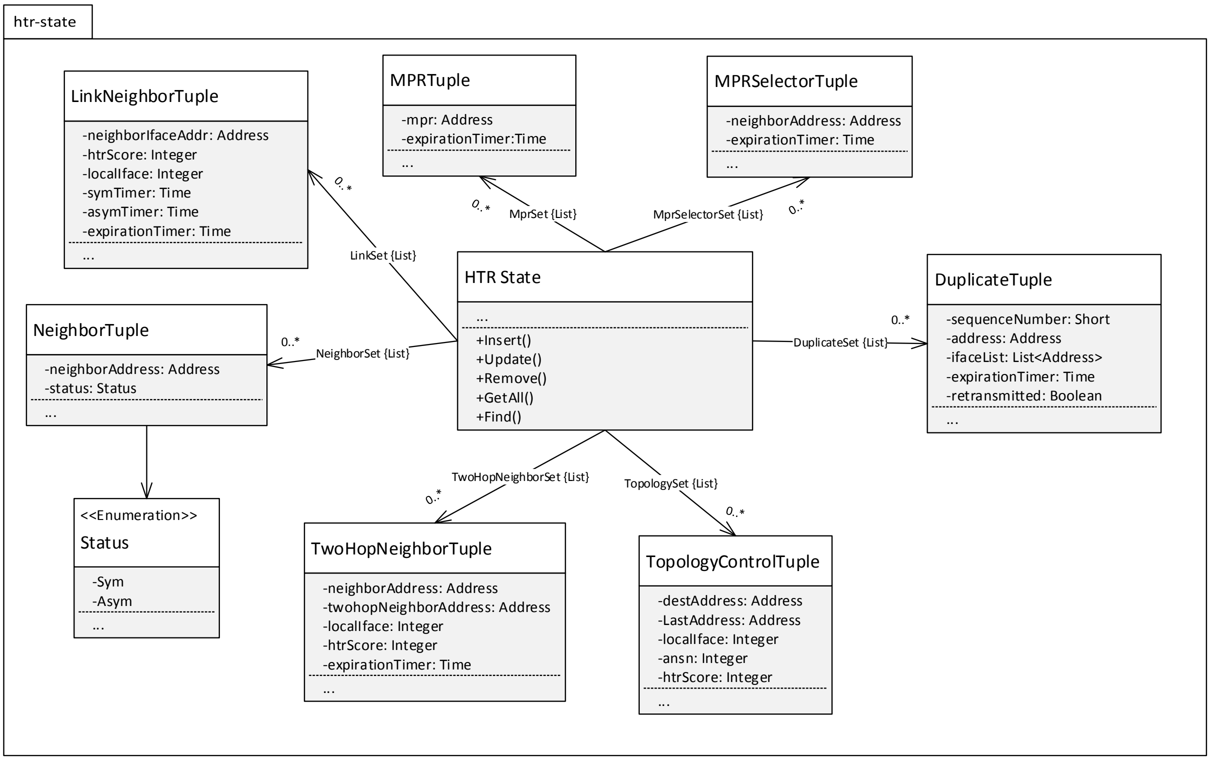


Figure 22: UML Class diagram for HTR State

The *HTR Header* component is the one to generate, manipulate, serialize and de-serialize the packet and messages of the HTR control message system. All classes defined in the HTR Header component derive from the ns-3 Header class. The *HTR Header* component models the HTR packet and messages (see Sections 3.6 and 3.7) and includes the following classes: *PacketHeader*, *MessageHeader*, *Hello*, *Link*, *Data*, *TC,* and *MessageType*. The *PacketHeader* class is the one representing the HTR Packet Header, the *MessageHeader* represents the HTR message header, the *Hello* class represents the HELLO message and includes a set of *Link* objects which are the ones to model each Link Message contained in the HELLO message. Each *Link* object contains a set of *Data* objects which represent the Data field of HELLO and TC messages, the *TC* class represents the TC message and also includes a set of Data objects. Figure 23 shows the UML class diagram for the HTR Header component.

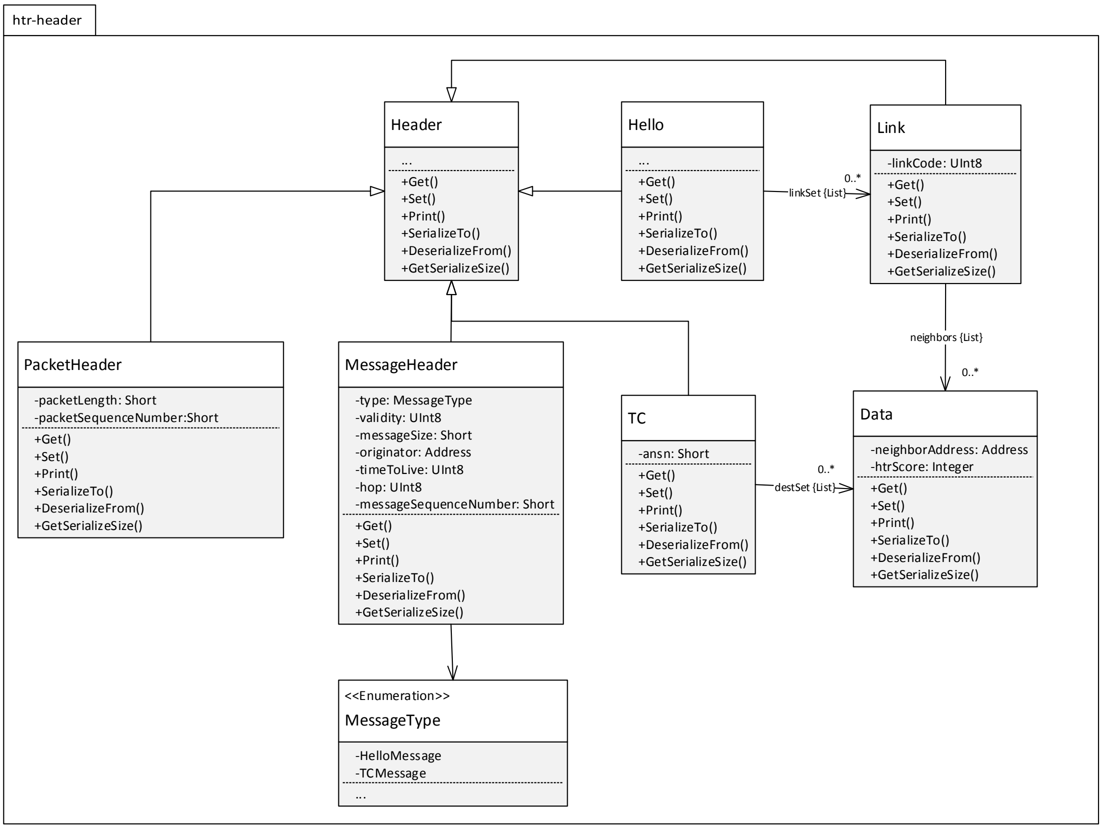


Figure 23: UML class diagram for HTR Header

The *HTR Routing* component is the controller of the HTR protocol, and is the one to control the sending (i.e. *sendHello* and *sendTc* methods), forwarding (i.e. *forwardControlMessage* method) and processing of control messages (i.e. *receivePacket*, *processHello* and *processTc* methods), to maintain the states of its repositories using the functions (add, retrieve, modify, destroy) provided by the *HTR State* component and the *removeExpiredStates* method that also uses this component to destroy expired records of the repositories, to trigger the MPR computation (i.e. *computeMprSet* method), path computation (i.e. *computeRoutingTable*) that uses the *MultipathDijkstra* algorithm, and to control the sending (i.e. *RouteOutput* method), forwarding and delivering (i.e. both using *RouteInput* method) of each data packet.

To compute the HTRScore, the *HTR Routing* component uses the *calculateHtrScore* method which uses, as supporting methods, the *estimateEij*, *getCurrentyEnergy*, *getInitialEnergy* methods and also the link stability information (also used for HTRScore calculation) provided by the *HTR Device Info* objects. The first three methods (i.e. *estimateEij*, *getCurrentyEnergy* and *getInitialEnergy*) use the information provided by the device energy models (see Section 4.3), with some of these developed during this work. The *HTR Device Info* class is the one for calculating the probability of loss of packets (link stability used on HTRScore) for each ns-3 *NetDevice* (i.e. *WifiNetDevice*, *WimaxNetDevice*, *LteNetDevice*) that it is tracking (i.e. installed).

The *HTR Routing* component provides configuration parameters, in order to control the behavior of some of its internal mechanisms, that are the following: *alpha*, *beta*, *gama* and *theta* that are used for the HTRScore computation, the *multipath* parameter that indicates the number of alternative routes for the *MultipathDijkstra* to compute, the *helloInterval* and *tcInterval* that defines the HELLO and TC sending intervals respectively. After the *MultipathDijkstra* path computation, the HTR Routing controller stores a map (dictionary) data structure (*RoutingTableMultipathMap* class) where the key for it is the address for each destination computed and the data value for each correspondent key is a set, with size equal to the *multipath* parameter, containing a list of routes to reach the destination. Each route is recorded as a RoutingTableEntry object and this one is used to generate the Ipv4Route object during the *RouteInput* and *RouteOutput* methods. The *GetRoutingTableEntries* method, used by the *HTR Pyviz plugin* (see Section 4.4) to retrieve and show computed routes, is used to return all routes computed by the path computation mechanism. Figure 24 shows the UML class diagram for the *HTR Routing component*, and Figures Figure 25 and Figure 26 show the UML sequence diagram for the processing mechanism of HTR control messages.

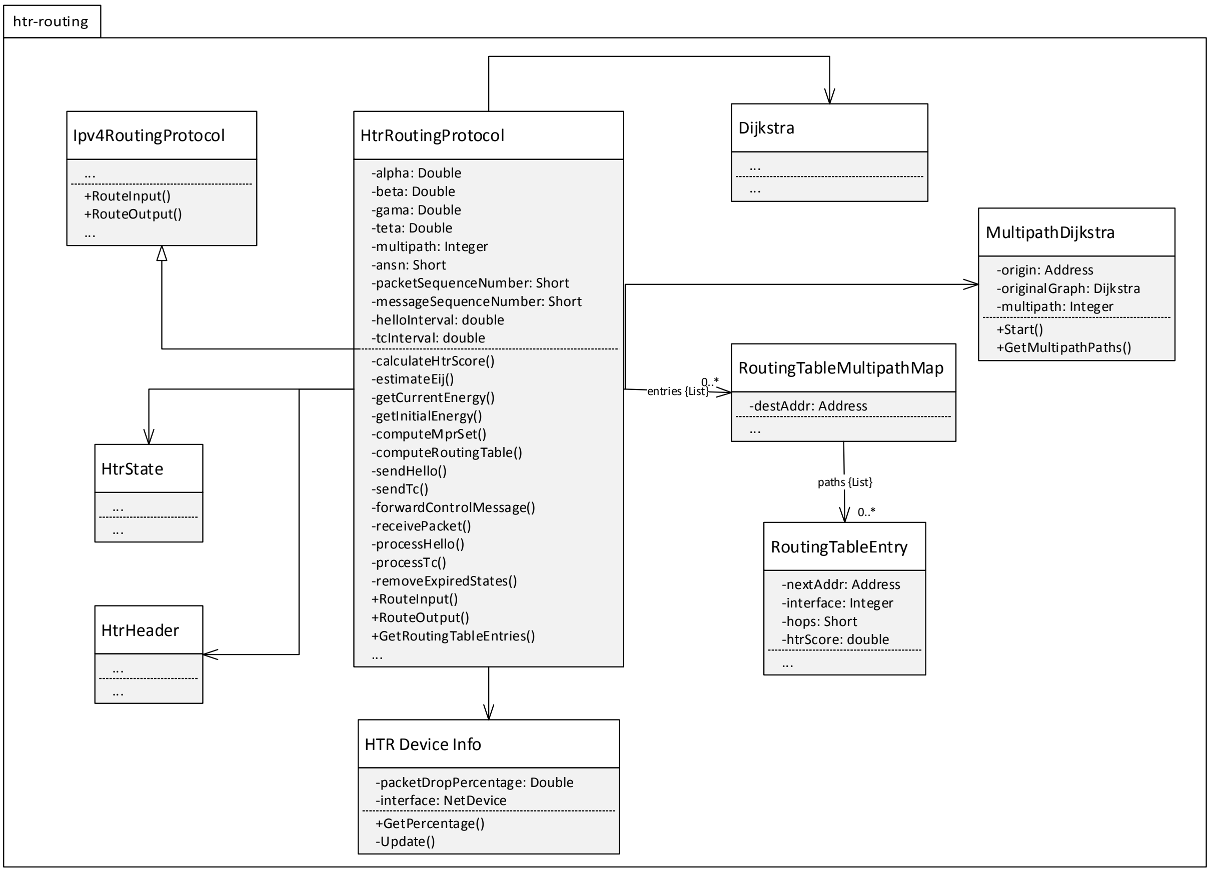


Figure 24: UML Class diagram for HTR Routing

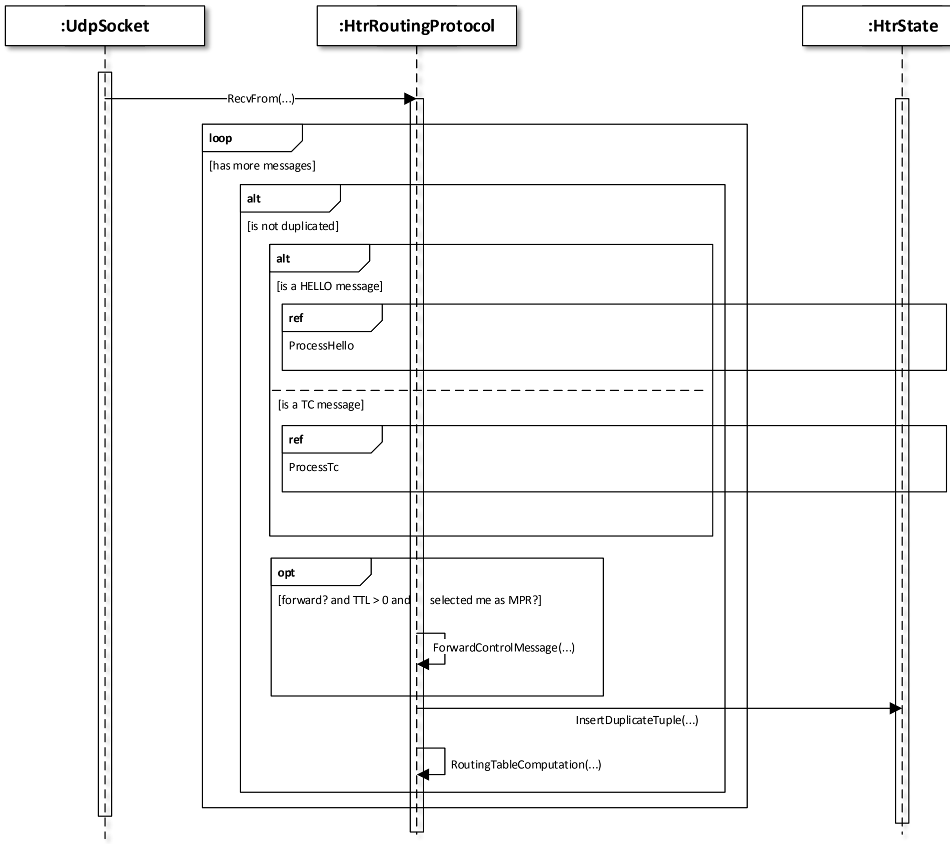


Figure 25: UML Sequence diagram for receiving of HELLO and TC messages processing

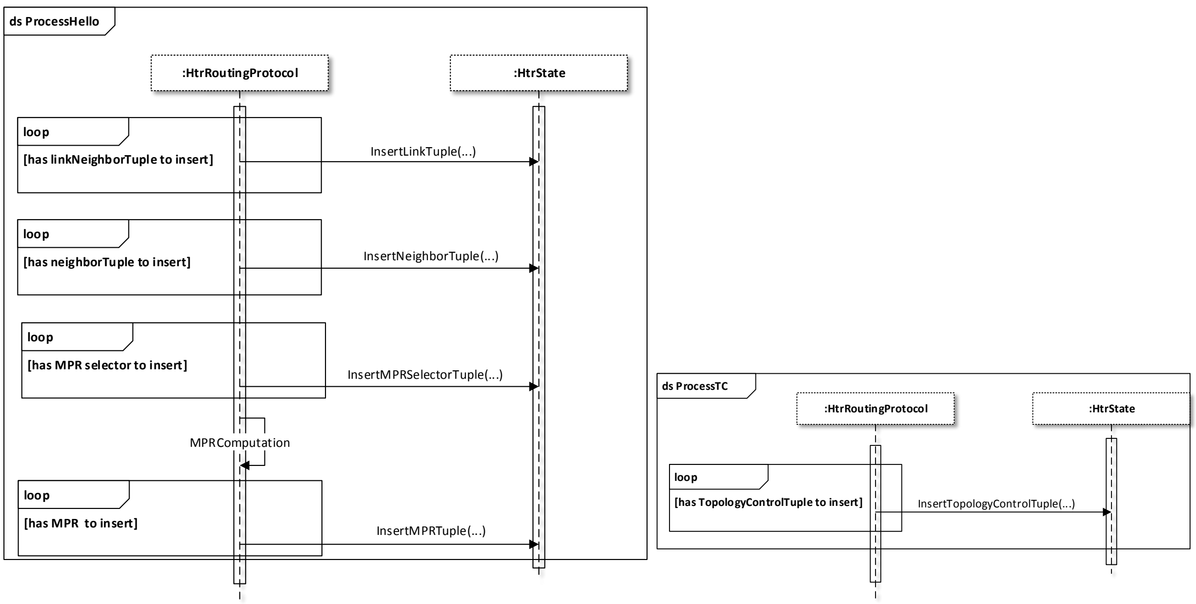


Figure 26: UML Sequence diagram for Process Hello and Process TC

4.3 ENERGY FRAMEWORK

Since energy consumption is a key issue for wireless devices, researchers often need to investigate the energy consumption of a battery powered node or the overall network, while running network simulations. This requires the underlying simulator to support energy consumption and energy source modeling. The ns-3 provides an energy framework consisting of a set of energy sources, a set of device energy models, and the interfaces interconnecting them. Figure 27 represents the energy framework with its core components as well as its new components, highlighted in black, which were developed during the course of this dissertation and enable research on the energy consumption of a network element using LTE or WiMAX devices. Actually, the energy framework just includes support for IEEE 802.11 devices. Those components are described during this section.

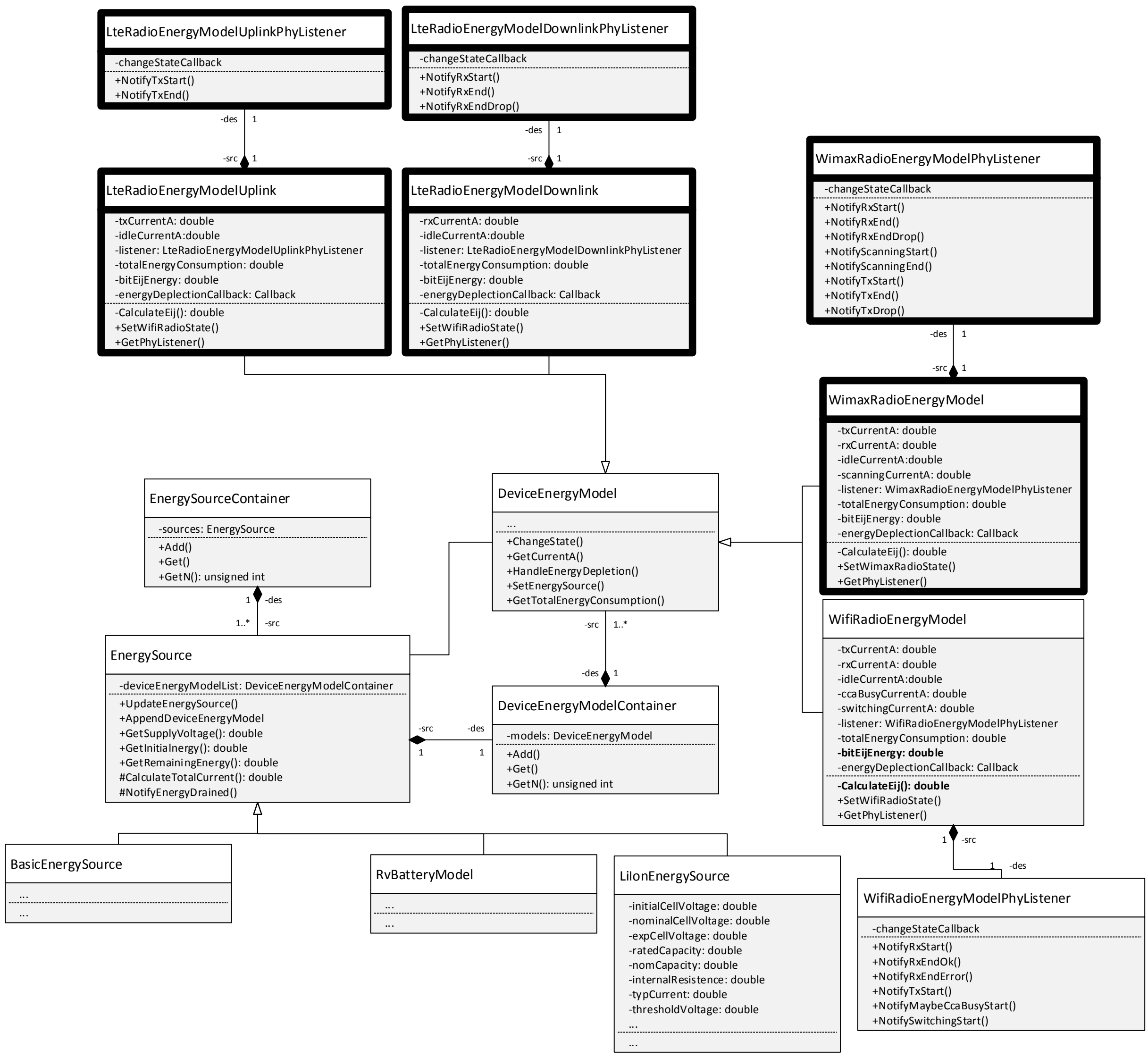


Figure 27: UML Class Diagram for Energy Framework

4.3.1 ENERGYSOURCE

EnergySource models the power supplies or batteries of network nodes, and includes linear as well as nonlinear battery models. The linear energy source has a user specified capacity in Joules and discharges linearly towards zero. They are easy to set up and configure, but do not capture crucial aspects of real life batteries [103]. On the other hand, non-linear battery models are more difficult to configure and are typically more accurate since they model the discharge curves of specific batteries.

Real-life batteries exhibit nonlinear effects, namely, the *Rate Capacity Effect* and the *Recovery Effect*. The first represents the decrease in battery lifetime when the current draw is higher than the rated value of the battery. The second represents the increase in battery lifetime when the battery is alternating between discharge and idle states. Ignoring these effects in battery models may lead to incorrect simulation conclusions [104]. For example, using a linear energy model in simulation implies that switching the radio between operating and idle states leads to energy consumption that is not verified, if *Recovery Effect* is considered [105]. Further, the reduced battery life caused by the *Rate Capacity Effect* is not captured by the linear battery model. Therefore, it is important for battery models to capture both the *Rate Capacity Effect* and the *Recovery Effect*.

These effects are captured by several nonlinear models proposed in the literature [104], which are classified as electrochemical, stochastic, electrical circuit and analytical. Electrochemical models are the most accurate but are also computationally intensive; stochastic models require building look up tables for battery non-linearity, making them cumbersome to use; circuit models use circuit simulation tools which are difficult to integrate with network simulation tools; and analytical models [106] use battery discharge curves to construct mathematical expressions of battery behavior. Analytical models are accurate, computationally efficient, easy to configure and are the ones adopted in the ns-3 energy framework for non-linear models.

Actually, the ns-3 energy framework includes three main types of energy sources derived from the *EnergySource* class: *BasicEnergySource*, *RvBatteryModel* and *LiIonEnergySource*. The *BasicEnergySource* is an ideal and linear energy source. The *RvBatteryModel* is a nonlinear battery model that implements the *RakhmatovVrudhula* (R-V) battery model [106], [107]. The *LiIonEnergySource* models a generic lithium Ion (Li-Ion) battery based on [108]–[110]. Both non-linear models are capable of modeling the *Rate Capacity Effect* and the *Recovery Effect*.

During the development and validation of this dissertation, the *LiIonEnergySource* was used as the battery source. The model used for this one can be fitted to any type of Li-Ion Battery simply by changing the model parameters. The default values of those parameters are fitted based on the *Panasonic CGR18650DA* Li-Ion battery [111]. The model requires several parameters to approximate the discharge curves (see Table 9):

Table 9: EnergySource Model Parameters

Parameter	**Description**
InitialCellVoltage	Maximum voltage of the fully charged cell.
NominalCellVoltage	Nominal cell voltage used to determine the end of the nominal zone.
ExpCellVoltage	Cell voltage at the end of exponential zone.
RatedCapacity	Rated capacity of the cell in Ah (ampere-hour).
NomCapacity	Cell capacity (Ah) at the end of the nominal zone.
ExCapacity	Cell capacity (Ah) at the end of the exponential zone.
InternalResistance	Internal resistance of the cell in Ohms.
TypCurrent	Typical discharge current value used during the fitting process.
ThresholdVoltage	Minimum threshold voltage below which the cell is considered depleted.

For instance, a discharge curve [108] is show in Figure 28 and the nominal and exponential zones are highlighted. Those zones are essential to extracting the required parameters.

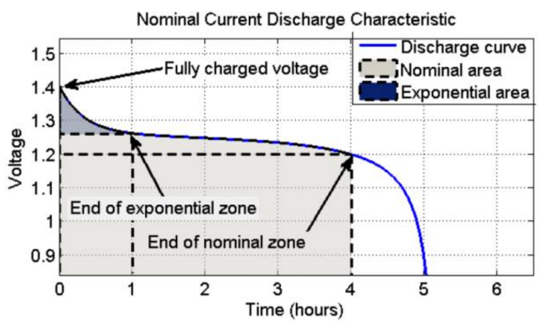


Figure 28: Example of Discharge Curve for a Li Ion Non-Linear Battery

In order to incorporate the *Rate Capacity Effect*, one uses supply voltage and current draw from all devices to calculate energy consumption. It polls all devices on the same node periodically to calculate the total current. The total current draw is calculated as the sum of the current draw from all *DeviceEnergyModel* objects attached to the same *EnergySource*. Connecting an energy source to a device energy model implies that the corresponding device draws power from this source. A node can have one or more energy sources and each one can be connected to multiple device energy models. Moreover, the energy source stores an initial amount of energy defined during the scenario elaboration or using the default value defined in its model.

Each node can access the *EnergySource* objects for information, such as the remaining energy (in Joules) or the energy fraction (i.e. battery level). This enables the implementation of energy aware protocols in ns-3, such as the HTR protocol. The *Energy Source* API is defined as follows in Table 10:

Table 10: EnergySource API Methods

Method	Description
GetInitialEnergy	Returns the initial energy stored in the energy source.
GetRemainingEnergy	Returns the remaining energy in the energy source.
GetEnergyFraction	Returns the energy fraction of EnergySource. Energy fraction is defined as (Remaining Energy ÷ Initial Energy).
UpdateEnergySource	Used by DeviceEnergyModel objects to notify the EnergySource to update its remaining energy when state change has occurred in the devices. The EnergySource can also schedule periodic calls of this function to constantly update the remaining energy values. The exact method used for the update depends on the energy source model being implemented. The LiIonEnergySource update process is defined in [109].
AppendDeviceEnergy Model	Used to add a DeviceEnergyModel to the list of DeviceEnergyModel objects kept within the EnergySource object.
CalculateTotalCurrent	This method queries all DeviceEnergyModel objects attached to the EnergySource for their current draw and then calculates the total current drawn for the energy source.

4.3.2 DEVICEENERGYMODEL

The *DeviceEnergyModel* is the energy consumption model of a device. It is designed to be a state based model where each device is assumed to have a number of states, and each state is associated with a power consumption value. A *DeviceEnergyModel* must communicate the new current draw of the device to the *EnergySource* whenever the state of the device changes. The Energy Source will then calculate the new total current draw and update the remaining energy.

Actually, the energy framework just includes a *DeviceEnergyModel* that only supports the WiFi physical layer (i.e. the *WifiNetDevice* is the only *NetDevice* supported). As a result of that, two new models were developed during this dissertation to support WiMAX and LTE devices. The *DeviceEnergyModel* objects are maintained as a list called *DeviceEnergyModelContainer* within the *EnergySource* object. The *DeviceEnergyModel* is defined as an abstract base class and its child classes are characterized by the actual devices they are modeling.

Devices such as WiFi, WiMAX and LTE radio have several operating states with different energy consumption accurate. Therefore, the *DeviceEnergyModel* is designed to be state-based, with a current draw value associated with each of the states. Using the current draw to represent each state enables the computation of the total current on an *EnergySource*, which is required for nonlinear battery models. A *DeviceEnergyModel* also allows the user to specify the behavior of the device when node energy is completely drained, which is implemented using a callback method defined in the child classes of the *DeviceEnergyModel*. When energy is completely drained, the callback will be invoked automatically.

For example, the callback method can be set to terminate the simulation or disable the device radio of the node. Similarly, it can be set to stop or remove the node from the rest of the network. By default, a warning message showing that energy is completely drained is shown if this callback is not explicitly set. During the development of this work, a stop function was developed that immobilizes the movement of the node and disables all of the devices and applications contained within it.

The *DeviceEnergyModel* API is defined as follows in Table 11:

Table 11: DeviceEnergyModel API Methods

Method	Description
SetEnergySource	Specifies the EnergySource object whereby the device model consumes energy.
ChangeState	Notifies the DeviceEnergyModel objects of changes in the state of the corresponding devices.
GetCurrentA	Returns the current draw (in Amperes) of the device at its present state.
HandleEnergyDepletion	EnergySource invokes this function when energy is completely depleted.
GetTotalEnergyConsumption	Returns the total energy consumed by the device. If called during simulation, it will return the total energy consumed by the device till the instant the function is called.

To introduce a new device model using this set of APIs, one needs to specify the operating states of the device and the current draw associated with those operating states. The energy framework already includes a child class nominated as *WifiRadioEnergyModel* to represent the energy consumption of a WiFi radio. To support WiMAX and LTE devices, this work implemented two new device energy models derived from the *DeviceEnergyModel* class: *WimaxRadioEnergyModel* and *LteRadioEnergyMode* (divided in two components).

4.3.3 WIFI ENERGY MODEL

The *WifiRadioEnergyModel* stores the current draw values for each WiFi radio operation state defined in the ns-3 *WifiPhy* class: TX, RX, IDLE, CCA BUSY and SWITCHING. The *WifiPhy* class represents the IEEE 802.11 [18], [19], [112] physical layer. By default, CCA BUSY and SWITCHING have the same current draw as the IDLE state. A similar state definition is also used in [113], [114]. Default current draws for these states are based on the Texas Instruments CC2420 radio chip datasheet [115], with supply voltage as 2.5V and currents as 17.4 mA (TX), 19.7 mA (RX), 0.426 mA (IDLE, CCA BUSY and SWITCHING) for its states respectively.

The ns-3 WiFi module already includes an element, designated as *WifiPhyListener*, which tracks the state of the WiFi radio. To help integrate the *WifiRadioEnergyModel* with the existing ns-3 WiFi module, the energy framework has implemented a listener based on the *WifiPhyListener* that notifies the *WifiRadioEnergyModel* whenever the WiFi radio state changes.

4.3.4 WIMAX ENERGY MODEL

The WiMAX device energy model, defined as *WimaxRadioEnergyModel* and developed during the elaboration of this work, uses the same design used in *WifiRadioEnergyModel*. However, the ns-3 WiMAX module does not include a class object to track its physical layer state. Instead, it uses the ns-3 tracing system [116], which is built on the concepts of independent tracing sources and tracing sinks, and a uniform mechanism for connecting sources to sinks. Trace sources are entities that can signal events that happen in a simulation and they provide access to interesting underlying data. Trace sinks are consumers of the events and of the data provided by the trace sources. For this model, the trace source is the WiMAX physical layer as it signals its actual state and the trace sink, denoted as *WimaxRadioEnergyModelPhyListener*, notifies the *WimaxRadioEnergyModel* whenever the state of the WiMAX physical layer changes.

The ns-3 WiMAX physical layer defines the following states: TX, RX, IDLE and SCANNING. The *WimaxRadioEnergyModel* power consumption is based on the Atmel AT86RF535B radio chip datasheet [117] that defines 3.3V for supply voltage and current values of 315 mA, 270 mA, 2.5 mA, 135 mA for those states respectively.

4.3.5 LTE ENERGY MODEL

The LTE device energy model is divided in two main models: *LteRadioEnergyModelDownlink* and *LteRadioEnergyModelUplink*. The separation occurs since the ns-3 LTE physical layer model is composed of two independent physical layer components, one for managing downlink and the other for uplink connections. The *LteRadioEnergyModelDownlink* handles the state and stores current draw values for the downlink component. Similarly for the LTE uplink physical layer element, the *LteRadioEnergyModelUplink* is used. The LTE defines the states TX and IDLE for uplink and RX and IDLE for the downlink physical layer components. The LTE physical layer uses the ns-3 tracing system to signal the current states of its downlink and uplink components independently. To help those models, the *LteRadioEnergyModelDownlinkPhyListener* and *LteRadioEnergyModelUplinkPhyListener* are used as trace sink to detect state changes and to notify the corresponding device energy models respectively.

The LTE device energy models are based on the Infineon BGA777L7 radio chip datasheet [118] that defines 2.8V for supply voltage and a 4.2 mA current for TX and RX states. Further, it defines a 0.53 mA current for the IDLE state.

4.3.6 HTR SCORE e_{ij}

To calculate the HTR Score, for the metric used by the HTR routing protocol and explained in Section 3.4, the e_{ij} must be calculated. The e_{ij} as described before is the transmission energy required for node i to transmit an information unit to its neighboring node j . Each device energy model described was modified and includes a function that estimates the e_{ij} parameter. The estimation calculates the energy consumption in joules to transmit one bit of information based on the total bytes sent and the total energy consumed during a TX state.

4.4 PYVIZ

The ns-3 simulator includes a live simulation visualizer, designated as PyViz [119], useful for debugging purposes such as examining whether the mobility model is working as expected, determining where packets are being dropped, visualizing spatial node distribution and so on. Although it is mostly written in Python, it works both with Python and pure C++ ns-3 simulations. The PyViz uses Gtk+ [120], PyGtk [121] and GooCanvas [122] for its graphical user interface (GUI) part and also accesses the ns-3 API directly via respective Python bindings. To add support for PyViz, the scenario must add the visualizer module as a dependency to its program (See [119] for details) and it uses the vis waf option while invoking the waf run command (see Appendix A)

4.4.1 HTR PYVIZ PLUGIN

Python modules appearing in *"contrib/visualizer/plugins/"* are loaded automatically by PyViz as plugins. Each plugin module must define a register function called as soon as the visualizer is initialized, passing the Visualizer class instance as sole argument. The Visualizer class is a singleton [123] that controls the whole visualizer GUI, and contains a list of nodes, channels, canvas, and arrows used to represent the packet transmissions and it also contains assorted GUI elements such as the play button.

Each plugin is written in the Python language and uses PyGtk library for creating graphical user interfaces. The Figures Figure 29 and Figure 30 demonstrate the HTR PyViz plugin, which includes, for each node, an option to show the current HTR Routing Table. The HTR Routing Table includes the following information: Destination Address, Next Hop, Interface (i.e output interface to reach next hop), Num. Hops (i.e. number of hops to reach destination) and HTR Score.

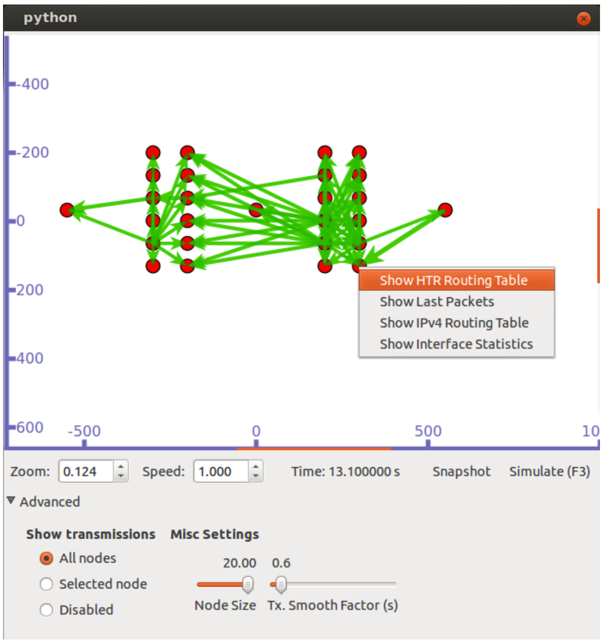
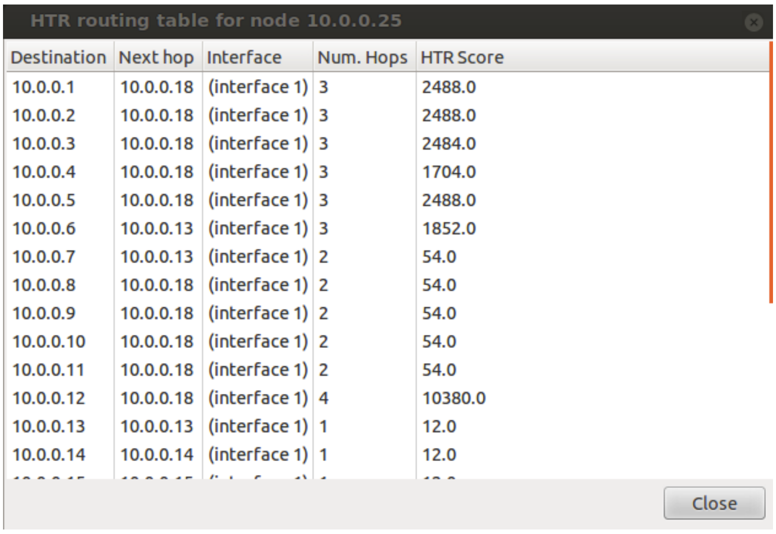


Figure 29: PyViz simulation and HTR Routing Table option for each node



HTR routing table for node 10.0.0.25

Destination	Next hop	Interface	Num. Hops	HTR Score
10.0.0.1	10.0.0.18	(interface 1)	3	2488.0
10.0.0.2	10.0.0.18	(interface 1)	3	2488.0
10.0.0.3	10.0.0.18	(interface 1)	3	2484.0
10.0.0.4	10.0.0.18	(interface 1)	3	1704.0
10.0.0.5	10.0.0.18	(interface 1)	3	2488.0
10.0.0.6	10.0.0.13	(interface 1)	3	1852.0
10.0.0.7	10.0.0.13	(interface 1)	2	54.0
10.0.0.8	10.0.0.18	(interface 1)	2	54.0
10.0.0.9	10.0.0.18	(interface 1)	2	54.0
10.0.0.10	10.0.0.18	(interface 1)	2	54.0
10.0.0.11	10.0.0.18	(interface 1)	2	54.0
10.0.0.12	10.0.0.18	(interface 1)	4	10380.0
10.0.0.13	10.0.0.13	(interface 1)	1	12.0
10.0.0.14	10.0.0.14	(interface 1)	1	12.0

Close

Figure 30: PyViz HTR Plugin

4.5 CHAPTER SUMMARY

In this chapter, initially an introduction for the open source widespread network simulators that are in use today was given and also an evaluation that determined that the ns-3 simulator was the appropriate network simulator to be used for this work. Next, the details of the implementation of the HTR protocol in the ns-3 simulator were provided and the extensions required for the operation of the protocol were also shown.

CHAPTER 5 PERFORMANCE EVALUATION

In this chapter, the proposal is evaluated. The chapter is organized as follows: Section 5.1 describes the evaluation methodology. Section 5.2, shows the details of the results, the initial conclusions and the lessons learned. There are three different ways to evaluate and compare the performance of MANET protocols [29]. The first one is based on mathematical model analysis, such as the one in [124], which uses parameters such as time and communication complexity for performance evaluation. In the second method, routing performance is analyzed based on simulation results, such as shown in [125], [126], using a network simulator such as ns-2, ns-3, and OMNET++, among others. The last method is the real implementation of routing protocols to analyze the performance using data from real-world implementations like the one in [10]. For this work, the second approach was used with the ns-3 simulator to collect the metrics and simulate the scenarios.

5.1 METHODOLOGY

The methodology section is divided into six parts. First, the metrics and data collected are explained. Second, the statistics methodology applied is discussed. Third, the heterogeneous technologies and their parameters are detailed. In the fourth, the propagation loss models used and their parameters are shown. In the fifth, the energy framework configuration is provided. Finally, the scenarios are discussed.

5.1.1 METRICS

It is useful to track several quantitative metrics that illuminate the internal efficiency of a routing protocol. The Internet Engineering Task Force (IETF) group elaborated an RFC document (RFC 2501) [127] that provides routing performance evaluation considerations, and which defines a list of quantitative metrics that can be used to assess the performance of any MANET routing protocol. Some of these considerations are taken into account in this project, together with some other metrics not covered by [127], which are well-known network metrics, and important for comparing this work proposal with the OLSR and the baseline HTR. This subsection is divided into two parts: one describing the metrics in use for scenario A and the other describing the metrics in use for scenario B; both scenarios are described in this section. ns-3 provides a native tool developed in order to help collect network metrics called *FlowMonitor*, introduced in [128]. The *FlowMonitor* tool was used in this work to collect the goodput, average end-to-end delay, and packet loss ratio. For the Expended Energy metric, the ns-3 energy framework and its enhancements proposed in this work were used to collect the states of the batteries. Finally, for the Convergence Time (CT), trace sinks and trace sources (using the ns-3 tracing system) were used to track the convergence time of the routing protocols.

5.1.1.1 SCENARIO A

HTR HELLO interval: this metric evaluates the impact of varying the HELLO refresh interval on convergence time and energy consumption. The value computed by this metric, which is given in seconds, corresponds to the HELLO refresh interval that gives the minimum convergence time and energy consumption. The HTR HELLO interval is computed by varying the network size (the number of nodes) and the technology applied. To measure the impact of node density and to investigate the HELLO interval required for minimum convergence and associated energy consumption, this work varied the HELLO interval between 0.5 and 5 seconds increasing it by 0.25s steps.

Convergence Time (CT): is defined as the elapsed time necessary for the protocol to converge. That is, the duration (in seconds) taken by a routing protocol to provide, for each node, a path towards any other node of the network [129].

Expended Energy: some or all of the nodes in a MANET may rely on batteries or other exhaustible means for energy provision. For these nodes, the most important system design criterion for optimization may be energy conservation. The Expended Energy metric measures the energy consumed (discharge of the battery) during the bootstrap of the network (convergence time) and is used to evaluate the impact of varying the HELLO interval on the energy consumption within the whole network. This metric measures the discharge of the battery in milliampere-hours (mAh), which is the charge accumulated, exactly 3.6 coulombs (C), by a current of one milliamperere in one hour. The mAh is commonly used in stating the capacity of batteries (i.e. how much power that the battery source can still hold over time) for cell phones, laptops and other electronic equipment. The value computed (in mAh) by this metric is the sum of all differences between the initial charge (maximum charge) of the battery and the current charge (resilient charge) at the end of the simulation, for each node of the network, in order to compute the resultant discharge of the batteries in mAh for all nodes.

5.1.1.2 SCENARIO B

Goodput: is the average number of data bits delivered (in Kbps for this work), which measures the bit efficiency of delivering data within the network.

Expended Energy: it measures the energy consumption (discharge of the batteries) of all nodes during the whole scenario lifetime and is measured in mAh. The value computed (in mAh) by this metric is the sum of all differences between the initial charge (maximum charge) of the battery and the current charge (resilient charge) at the end of the simulation for each node in order to compute the resultant discharge of the batteries in mAh for all nodes.

Packet Loss Ratio (PLR): is the ratio percentage calculated using the total number of data packets lost in transit divided by the total number of data packets generated [130].

Average end-to-end delay: the end-to-end delay of a packet in a network is the time the packet takes to reach the destination after it leaves the source. The average end-to-end delay (in seconds for this work) is obtained by averaging the delays of all transmitted packets in the network [131].

5.1.2 STATISTICS METHODS

Statistics is the art and science of collecting, analyzing, presenting and interpreting data [132]. In order to provide good accuracy for the results collected in this work and to provide statistical support for the whole evaluation process, some statistical methods were used. The statistical methodology applied during this work was as follows:

For each metric, an initial sample with size 50 was collected. The size 50 was chosen based on the central limit theorem [133], which states that the sampling distribution of random samples of size *n* from a population can be approximated by a normal distribution as the sample size becomes larger, and based on statistical practice that assumes that, for most applications, the sampling distribution of the sample can be approximated by a normal distribution whenever the sample is size 30 or more. In cases where the population is highly skewed or outliers are present, a sample of size 50 may be needed [132]. Thus, this work used the size of 50 for the size of the initial sample as a precaution. After, the samples were tested for normality using the Shapiro-Wilk's W statistic test [134], which has become a standard for small sample sizes (up to 50 samples); among several tests for normality, it provided and was extended by [135] to provide a test of normality for samples with size up to 2000. In statistics, the Shapiro-Wilk's W statistic test uses a null hypothesis that the sample is normally distributed, which is rejected if the test statistic W is lower than W_α [136], where W_α is a tabulated critical point given in [134], considering the size of the sample and the significance level (α) [132] in use, this one being usually 0.05 for the Shapiro-Wilk's W test. For the evaluation, the significance level of 0.05 was used.

$$n = \frac{\left(z\alpha/_2\right)^2 \sigma^2}{E^2} \tag{2}$$

$$\bar{x} \pm E, E = z\alpha/_2 \frac{\sigma}{\sqrt{n}} \tag{3}$$

If the sampling distribution can be approximate to a normal distribution (not rejecting the null hypothesis for the Shapiro-Wilk's W test), the evaluation being held during this section uses the process described in section 8.3 of [132], which describes how to choose an adequate sample size large enough to provide a mean value ($\bar{x}$) with a desired margin of error (E) at the chosen confidence interval (1-α) using (2). In (2), n is the adequate sample size to be computed; σ is the standard deviation of the 50 samples already collected (used as a preliminary sample) and $z\alpha/_2$ is the cumulative probability value for the standard normal distribution when using the confidence interval (a 95% interval is considered in this work). Using the value computed by (2), we collected n samples that give the interval estimate for the mean $\bar{x}$ as expressed in (3). The margin of error is depends of the metric in use; however, the value was chosen as such because it gives a good accuracy, which is required when comparing values, as shown in Table 12.

Table 12: Error margin for each metric

Metric	Error margin
Convergence Time (CT)	0.1 s
Expended Energy	0.05 mAh
Goodput	10 kbps
Packet Loss Ratio (PRL)	1%
End-To-End average delay	0.1s

If the Shapiro-Wilk's W test rejects the hypothesis for the normality of the population, 100 samples are collected and the median, (value in the middle when the data is arranged in ascending order) is used as the value for each metric to be evaluated. Since the distribution of the population does not have a normal distribution, the Wilcoxon Signed-Rank test [137], a non-parametric method, is used to state that the medians of the populations collected for each evaluation case and used for comparing (e.g. OLSR and Multipath HTR) are not identical.

For the statistical computing realized as described here, the integrated suite R was used. R is a language and environment for statistical computing graphics, and is free software for facilitating data manipulation, statistical calculation, and graphical display [138].

5.1.3 HETEROGENEOUS TECHNOLOGIES

This subsection describes the configuration of the heterogeneous technologies used, as shown in Table 13. Three well-known heterogeneous technologies were used: Wi-Fi, WiMAX and LTE. Wi-Fi includes the IEEE 802.11 a/b/g/n standards for wireless local area networks [139]; the first three (a/b/g) are natively supported by the ns-3 simulator. Among the a/b/g standards, the IEEE 802.11g, with a wireless channel frequency of 2.4GHz, was used as the physical layer model for Wi-Fi devices, since it is the only one to allow a maximum data rate up to 54 Mbps, when using a quadrature amplitude modulation (QAM) with 64 points on the constellations (QAM-64) [140]–[142] and Orthogonal frequency-division multiplexing (OFDM) [143]. For the transmission power, Wi-Fi devices use 20 dBm, chosen based on the work provided in [144] for Scenario A, and a transmission power interval varying from 35-39 dBm for Scenario B; this transmission power interval was chosen to enable long-range communication for Wi-Fi devices contained in Scenario B. For WiMAX and LTE, the default configuration that includes the propagation loss model, the transmission power, and the channel frequency defined in ns-3, was used for this work. For WiMAX, the OFDM method using quadrature amplitude modulation (QAM) with only 16 constellations and ½ of the coding rate had to be used, considering the evaluation of the COST-Hata[145], [146] propagation loss model for WiMAX done in [147], in order to enable the communication of mobile nodes with a distance from the WiMAX base station ranging from 250m to 500m when using the COST-Hata propagation loss model.

Table 13: Technologies configuration

Parameters	Wi-Fi	WiMAX	LTE
Physical layer model	PHY 802.11g	PHY 802.16	PHY 3GPP LTE
Wireless channel frequency	2.4 GHz	5 GHz	1929-1980 Mhz (Uplink) 2110-2170 Mhz (Downlink)
Propagation loss model	Two Ray Ground	COST-Hata	Friis
Transmission power	20 dBm (Scenario A) 35-39 dBm (Scenario B)	30 dBm	10 dBm (UE) 30 dBm (eNb)
Modulation	OFDM 64 QAM	OFDM 16 QAM	OFDM QPSK 16 QAM 64 QAM
Coding rate	3/4	1/2	1/2 2/3 3/4

5.1.4 PROPAGATION LOSS MODELS

An important part of any wireless network simulation is the appropriate choice of the radio propagation loss model to be used to model the performance of a wireless network channel or set of channels [148]. The study of radio propagation is largely concerned with what happens with a propagated radio signal between the source and the destination. The source produces a signal (an electromagnetic plane wave) that is modulated onto the carrier frequency [149]–[151]. On its way to the receiver (at roughly the speed of the light), the signal reacts with any number of obstacles and then is induced on the receivers' antenna and demodulated. Obstacles in the environment cause the signal produced to be reflected, refracted, or diffracted (e.g. frequency shifted due to Doppler spreading [152]), that attenuate the power of the signal (through absorption) and cause scattering and secondary waves. Furthermore, since the antenna radiates its signal simultaneously in all directions, the signal can take many paths to the receiver, and each path may interact with the environment in a chaotic way and arrive at the receiver delayed by some amount. Delayed signals in phase with one another, produce constructive interference (raising the amplitude of the resulting receiving signal) and the ones out of phase produce destructive interference (canceling the amplitude of the resulting receiving signal). The spread of these delays is called delay spread and the resulting attenuation is called multipath fading. When this attenuation is caused by large unmoving obstacles, it is referred to as shadowing, slow-fading, or large-scale fading and when it is caused by small transient obstacles, and varies with time, it is called scattering, fast-fading or small scale fading.

$$P_{rx} = P_{tx} + G_{tx} + G_{rx} - PL \tag{4}$$

$$SNR = P_{rx} - \left(N + \sum_{j}^{n} I_j \right) \tag{5}$$

$$P_{tx} + G_{tx} + G_{rx} - PL \geq MDS(P_e) \tag{6}$$

Considering a transmission with a theoretical isotropic antenna (i.e. which evenly distributes power in all directions), the entire radio link can be summarized (isolated from sources of external noise in the environment) by the log-domain link budget equation [150] shown in (4), with P_{rx} and G_{rx} being the power (in dBm for P_{rx} and in dBi for the G_{rx}) received and the receiver's antenna gain in the direction of the transmitter, with P_{tx} and G_{tx} being the transmitter's radio power in dBm and transmitter's antenna gain in dBi, and with the PL term including all attenuation (in dB), described above, due to path loss. The link budget equation is the accounting of all the aggregate gains and losses (attenuations) of many competing signals from the transmitter [150], through the medium to the receiver in a telecommunication system. Modeling the PL term is needed in order to compute the signal strength of a wireless transmission at the receiving stations which, in turn, is required to determine whether or not each of the potential receivers can, in fact, receive the information without bit errors [148]. For a given receiver design and modulation scheme, there is a relationship between the Signal to Noise Ratio (SNR), defined as (5), and bit error rate [150]. In (5), N is the environment noise (e.g. thermal noise [153]) and I_j is the set of interferers (i.e. interference from other known transmitters). Using this relationship, a minimum detectable signal, as a function of the acceptable error rate, for a given ratio "*MDS(*P_e*)*" can be determined, where P_e is the probability of bit error. Thus, the probable receiver of the information can, in fact, determine whether or not it might receive the information based on the inequality expressed in (6).

$Friss\ path\ loss\ (PL) for\ isotropic\ antennas$:

$$PL = -10 \log \left[\frac{G_{tx} G_{rx} \lambda^2}{(4\pi)^2 d^2}\right] \tag{7}$$

$COST - Hata\ (PL)\ for\ urban\ environment$:

$$PL = 46.33 + 33.9 \log f_c - 13.82 \log h_b - a(h_m) + (44.9 - 6.55 \log h_b) \log d \tag{8}$$

$Hata\ a(h_m)$:

$$a(h_m) = (1.1 \log f_c - 0.7) h_m - (1.56 \log f_c - 0.8) \tag{9}$$

$\boldsymbol{Two\ Ray\ Ground}$:

$$PL = 40 \log d - (10 \log G_{tx} + 10 \log G_{rx} + 20 \log h_{tx} + 20 \log h_{rx}) \tag{10}$$

In ns-3, the COST-Hata is the default propagation loss for WiMAX as well as the Friis [154] that is defined by the LTE-EPC Network Simulator (LENA) project [155], which is the project implementing LTE for ns-3 as the default propagation model. The Friss propagation loss model proposes a formula for free-space transmission loss that predicts that received power decays as a function of the distance (d) in meters between the transmitter and receiver raised to some power (i.e. power law function) as show in (7), where λ is the wavelength of the carrier in meters, and G_{tx} and G_{rx} are the gains for the transmitter's and receptor's antennas respectively. The COST-Hata loss model is derived from the Hata model [156] and depends upon four parameters for the prediction of propagation loss that can be used in urban, suburban and rural environments: center frequency (f_c), height of received antenna (h_m), height of base station (h_b) and distance between base station and received antenna (d).In (8) and (9), the COST-Hata prediction equation for urban environments is provided; for other environments one can refer to [157], [158]. The Two Ray Ground propagation loss model, used for Wi-Fi in this work, is a modest extension to the Friss path loss model, which considers a second path that reflects from the ground between the transmitter and receiver and which was initially developed by Rappaport in [149].The formula for the Two Ray Ground propagation loss model is shown in (10), where h_{tx} and h_{rx} are the heights in meters for the transmitter's and receptor's antennas, d is the distance in meters between these antennas, and G_{tx} and G_{rx} are the gains for these antennas, respectively. For details of the propagation loss in use in this work (COST-Hata, Friss and Two Ray Ground), please refer to [148], [150]. For further details on the concepts of radio propagation, one can refer to [149], [151]. Table 14 shows the parameters used for the configuration of each of the propagation loss models in use.

Table 14: Propagation Loss Model Configuration

Propagation Model	Default Parameters	Values
COST-Hata	Center Frequency	2.3 GHz
	Base Station Height	50m
	Mobile Station Antenna Height	3m
	Minimum Distance	0.5m
Friis	Wave Length	58.25mm
	System Loss	1
	Minimum Distance	0.5m
Two Ray Ground	Wave Length	58.25mm
	System Loss	1
	Minimum Distance	0.5m
	Mobile Station Antenna Height	1m

5.1.5 ENERGY MODELS

Each node uses a power source based on the published Panasonic CGR18650DA [111] Li-Ion battery, and for each technology in use a correspondent model for its energy consumption was defined based on real radio chips [115], [117], [118]. Table 15 shows the parameters for the energy model of each device (see Section 4.4). These parameters include the supply voltage and current for each state of the device.

Table 15: Radio Chip Energy Models

Technology	Radio Chip	Parameters	Values
Wi-Fi	Texas Instruments CC2420	Supply voltage	2.5 V
		TX current	17.4 mA
		RX current	19.7 mA
		IDLE, CCA BUSY, SWITCHING current	0.426 mA
WiMAX	Atmel AT86RF535B	Supply voltage	3.3 V
		TX current	315 mA
		RX current	270 mA
		IDLE current	2.5 mA
		SCANNING current	135 mA
LTE	Infineon BGA777L7	Supply voltage	2.8 V
		TX current	4.2 mA
		RX current	4.2 mA
		IDLE current	0.53 mA

5.1.6 SCENARIOS

This subsection describes each scenario evaluated and is divided in two parts: Scenario A and Scenario B. Scenario A evaluates the performance of OLSR and Tuned HTR (HTR protocol when varying the HELLO interval) using the metrics HTR HELLO interval, Convergence Time, and Expended Energy. Scenario B evaluates the performance of the OLSR, HTR baseline and Multipath HTR (HTR proposed in this work) using the metrics Goodput, Expended Energy, Packet Loss Ratio and Average End-to-End delay.

5.1.6.1 SCENARIO A

The main characteristic of this scenario is to use heterogeneous technologies such as WiMAX and LTE combined with Wi-Fi. The selected topology is composed of a unique tower that stays at the center of the scenario, which could be either LTE or WiMAX technology depending on the test case. Also in the topology there is a group of nodes spread around the range that act as technology bridges (i.e. they have two distinct radio interfaces and could therefore route the messages between two or more radio technologies), and a border group of

Wi-Fi nodes. Each bridge spreads over the circle according to the Normal distribution and has a Wi-Fi interface. Furthermore, it stays in the center of another circle and communicates with a group of nodes using Wi-Fi devices that are also deployed using locations that follow the same distribution. Figure 31 illustrates the basic topology of the scenario. The number of bridges n, ranging from two to eight, was chosen in such a manner that a group of n^2, composed of Wi-Fi devices, can communicate with other groups using the bridge as a point of access. The simulation configuration details are shown in Table 16.

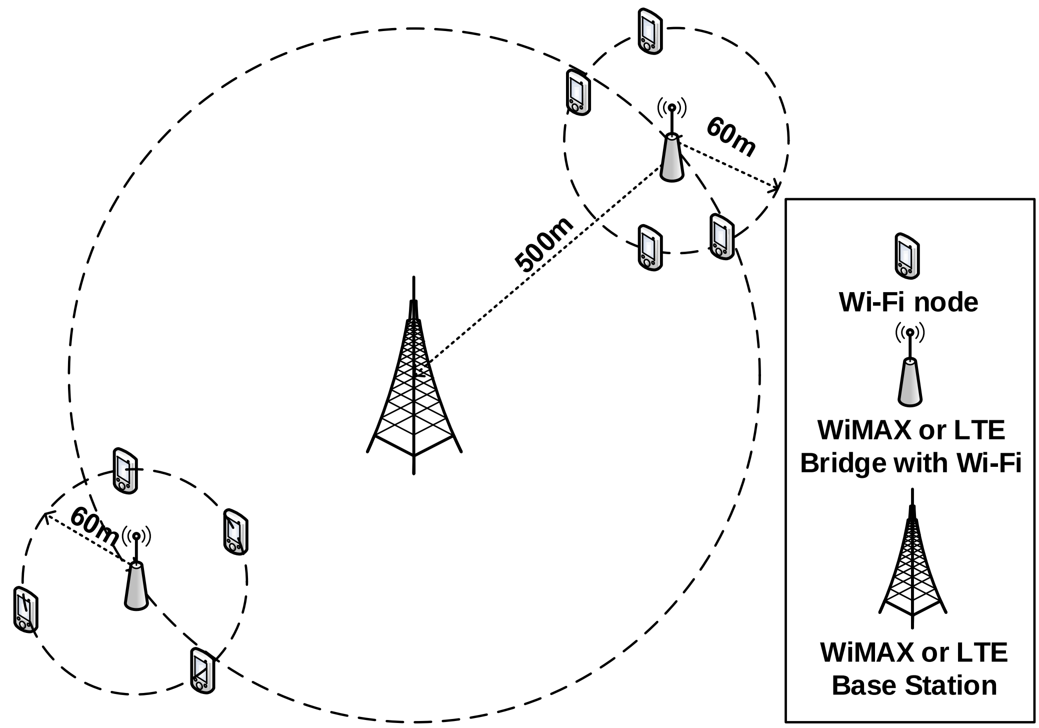


Figure 31: Simulated scenario A with heterogeneous technologies

Table 16: Configuration Parameters of Simulation Scenario A

Parameters	Values
Simulator	ns-3
Routing Protocol	OLSR / Tuned HTR
Simulation Area	1200m x 1200m
Max Convergence Time	30s
OLSR Hello Interval	2s
Tuned HTR Hello Interval	0.5-3.0s
TC Interval (OLSR / HTR)	5s
Energy level	100 mAh
Bridge density (LTE or WiMAX)	2-8
Wi-Fi density	4-64

5.1.6.2 SCENARIO B

Figure 32 shows the chosen topology scenario. It is composed of heterogeneous technologies and the main idea is to send traffic from nodes in the regions of the extremities (A). The traffic is generated, using a single constant bitrate (CBR) application generating data at 300 Kbps, 600 Kbps and 1Mbps. Each flow transmits data packets with size of 512 bytes and uses the UDP protocol as the transport protocol. The application has duration of 1000 seconds. Nodes in (A) and (B) have only Wi-Fi interfaces, nodes in the (C) area are bridges responsible for changing Wi-Fi to WiMAX or LTE and vice-versa. Finally, those nodes communicate with the other line of bridges through the tower (D). Therefore, the routes made between nodes in (A) have to pass through, at least, one node from (B), one node from (C), a tower (D), a node from (C), another node from (B) to, finally, arrive in destination node in (A). Thereby, it is possible to have several routes connecting extremity nodes.

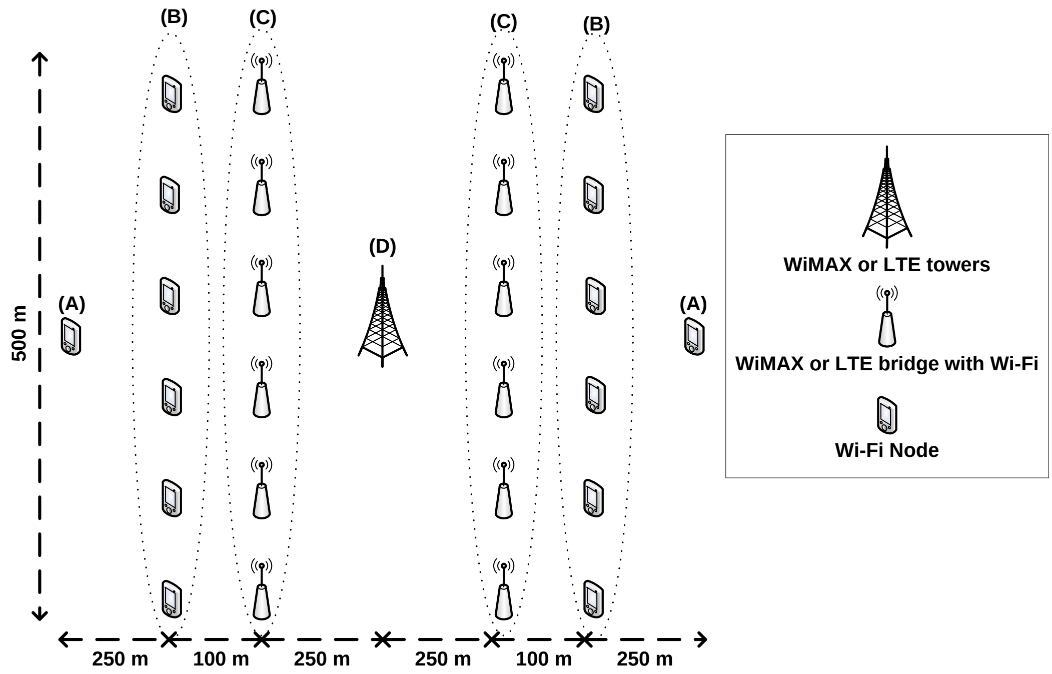


Figure 32: Simulated scenario B with heterogeneous technologies

To evaluate this scenario, this work used the configuration shown in Table 17. To ensure the heterogeneity of the network, the setup included three distinct technologies (Wi-Fi, WiMAX, and LTE) where two of them are used at each time, i.e., each protocol is executed in the scenario twice, one using Wi-Fi and WiMAX and a second time using Wi-Fi and LTE.

The initial energy level of each node was established in order to guarantee that, at the end of the simulation, all nodes will maintain at least 25% of resilient energy. This definition allows for better control of the HTRScore variation and prevents the deactivation of nodes.

Table 17: Configuration Parameters of Simulation Scenario B

Parameters	Values
Simulator	ns-3
Routing Protocol	OLSR / HTR / Multipath HTR
Simulation Area	1200m x 500m
Simulation Time	1010s
Applications	CBR
Application Packet Size	512 bytes
CBR start-end	10 – 1010s
Transport Protocol	UDP
Network Protocol	IPv4
IP Fragmentation Unit	2048
Data Rate	300 / 600 / 1000 Kbps

5.2 EXPERIMENTAL RESULTS

In this section, the experimental results for each scenario are presented. First, the results for scenario A are shown and afterward the results for scenario B and a discussion for this scenario are presented.

5.2.1 RESULTS FOR SCENARIO A

This subsection shows the experimental results for Scenario A.

5.2.1.1 HTR HELLO INTERVAL

First, this work examines the HTR performance in terms of two metrics: convergence time and energy consumption. This is done under different HELLO intervals and while also varying network density for the two heterogeneous scenarios, one using WiMAX and Wi-Fi and the other using LTE and Wi-Fi devices. As result, Figures Figure 33 and Figure 34 show the energy consumption and Figures Figure 35 and Figure 36 shows the convergence time. In those figures it can be seen that the best HELLO interval is between 2.0 and 2.5 seconds, which is the range that gives the best convergence interval and lowest energy expenditure.

Figure 37 depicts the best performance obtained while varying the HELLO interval for both scenarios. As it can be seen, the more nodes that are added to the network, the more frequently the HELLO messages should be sent. In addition, it also is observed that the HELLO interval for the WiMAX and Wi-Fi scenario had to increase the frequency faster than LTE and Wi-Fi case. Because an explanation for this fact cannot be found in the literature, this work decided not to speculate about that and to realize a more detailed evaluation (future work). For this, we are conducting new experiments using other propagation loss models as mentioned in the conclusion section.

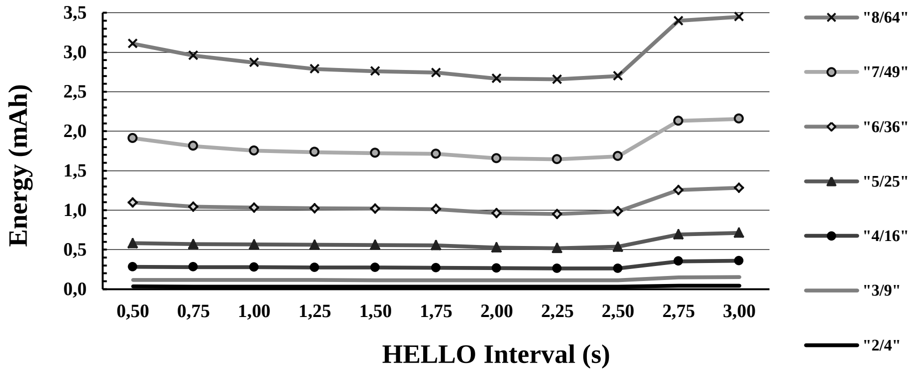


Figure 33: Energy Consumption using LTE and Wi-Fi technologies

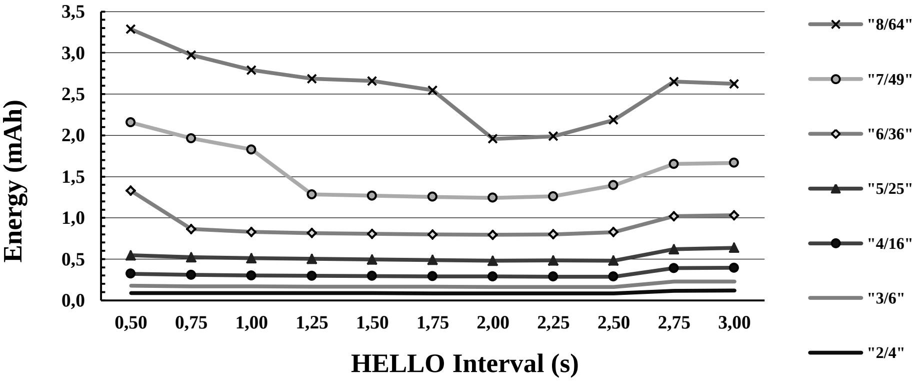


Figure 34: Energy Consumption using WiMAX and Wi-Fi technologies

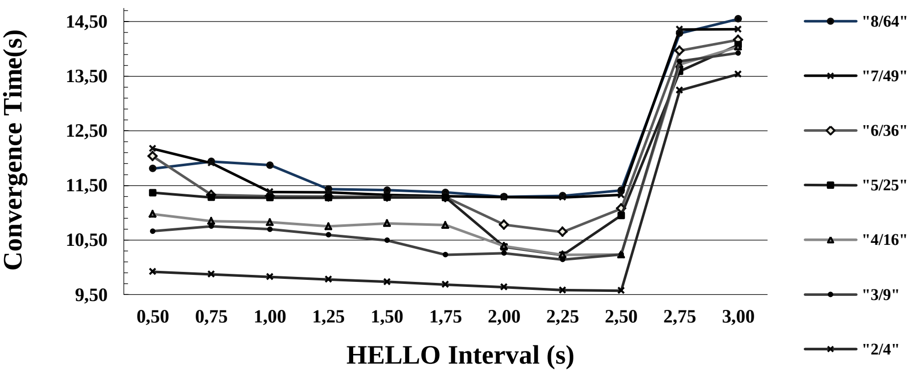


Figure 35: Convergence using LTE and Wi-Fi technologies

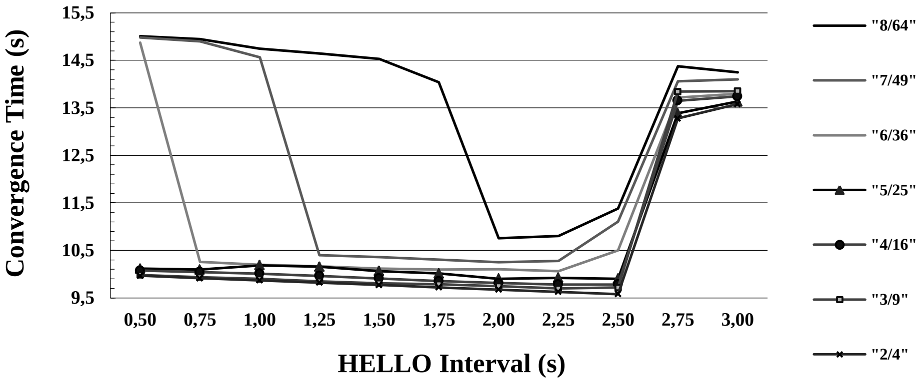


Figure 36: Convergence using WiMAX and Wi-Fi technologies

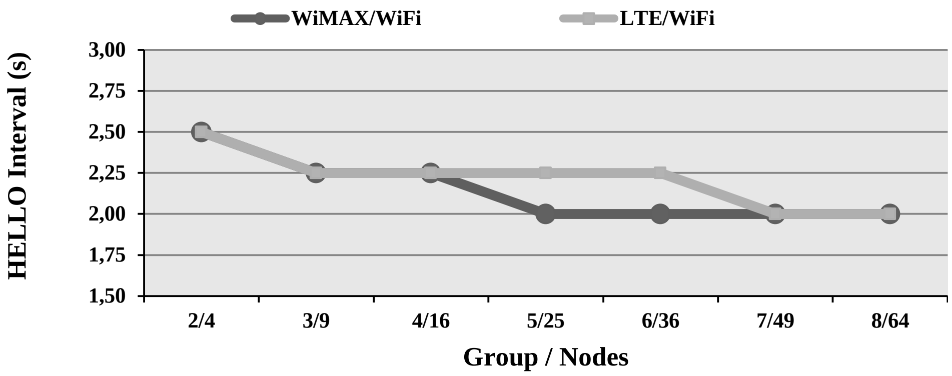


Figure 37: HTR HELLO interval for heterogeneous scenarios and using the tuned HTR

5.2.1.2 Convergence Time (CT)

The Convergence Time is defined as the elapsed time necessary for the protocol to converge. That is, the time taken by a routing protocol to provide, for each node, a path towards any other node. Using the HELLO interval defined by the previous metric, Figures Figure 38 and Figure 39 exhibit the Convergence Time for each case (density of nodes) and compares the tuned HTR with OLSR when using heterogeneous technologies.

Figure 38 shows that the OLSR protocol CT was almost 10% higher than the tuned HTR when using WiMAX and Wi-Fi. In Figure 39, the tuned HTR CT was almost 6% lower than OLSR when using LTE and Wi-Fi. In both cases, the collected results demonstrate that, with the increase of network density, more time for network convergence is needed.

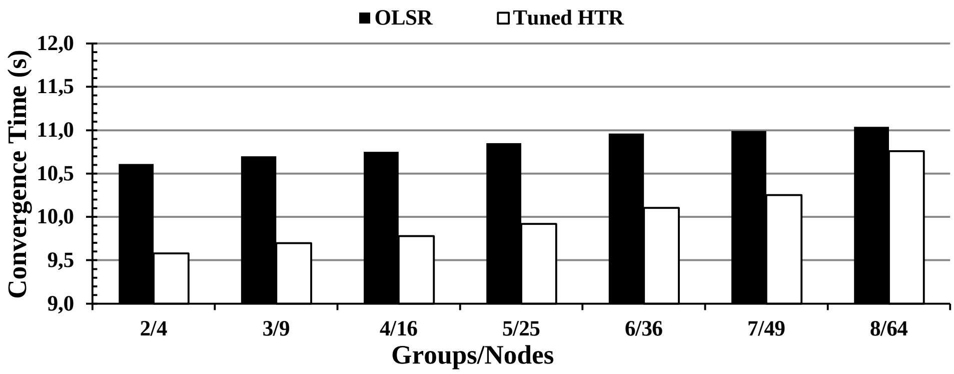


Figure 38: Comparing tuned HTR and OLSR convergence time using WiMAX and Wi-Fi devices

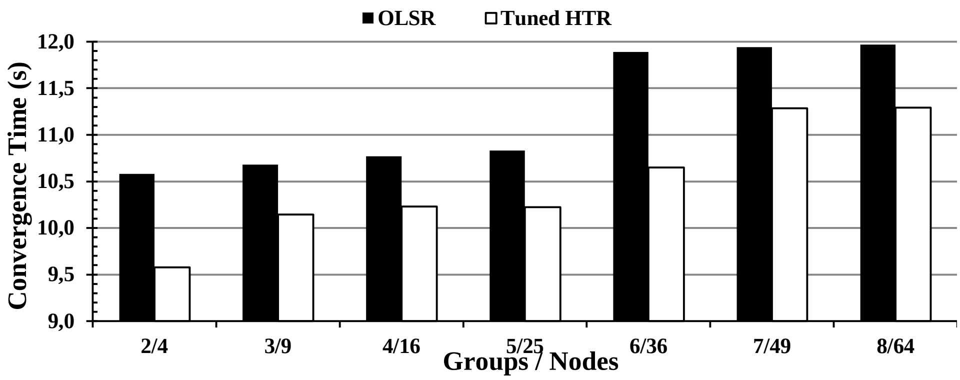


Figure 39: Comparing tuned HTR and OLSR convergence time using LTE and Wi-Fi devices

However, in order to investigate how specific wireless technologies affect the CT, Figure 40 shows a comparison between LTE and WiMAX. Please note that, using WiMAX technology, tuned HTR converges more quickly (approximately 4.5%) than when using LTE. In [148], the performance of a number of routing protocols, including AODV, DSDV [159] and OLSR, has been evaluated while subjected to well-known propagation loss models. One important result is that the COST-Hata (used with the WiMAX ns-3 implementation), outperforms the Friis propagation loss (used in LTE´s ns-3 implementation). Thus, results are in line with the results from [148], as shown in Figure 40.

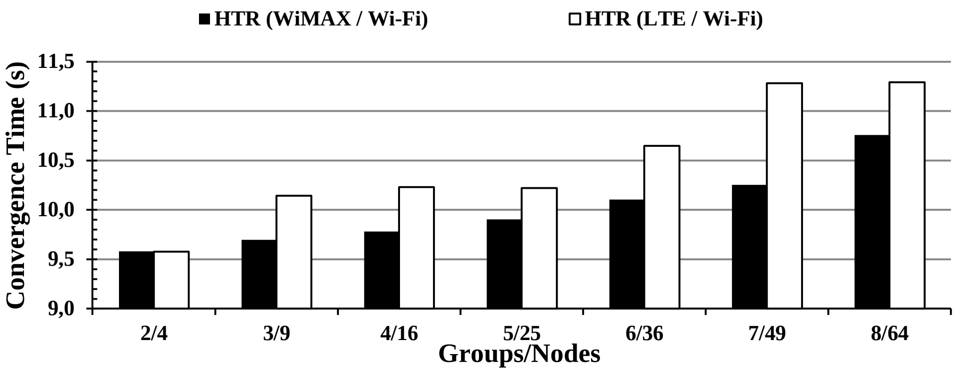


Figure 40: Comparing WiMAX and LTE using the convergence interval metric and tuned HTR

5.2.1.3 Expended Energy

Energy consumption usually varies according to the manufacturer of the device and the technology in use and must, therefore, be carefully analyzed in order to decrease the energy expenditure of the whole network. Figures Figure 41 and Figure 42 show a comparison of the Expended Energy during a given convergence interval between OLSR and the proposed tuned HTR.

Figure 41 shows the expended energy when using WiMAX and Wi-Fi. As it can be seen, the power expenditure when using OLSR is approximately 6% lower than tuned HTR. Figure 42 shows the Expended Energy when using LTE and Wi-Fi technologies and indicates that the tuned HTR energy expenditure was around 4% lower than OLSR.

In order to measure the impact on energy consumption by choosing different manufacturers and technology, Figure 43 shows additional results and compares LTE and WiMAX energy consumption when using tuned HTR. The results indicate that the power expenditure in LTE is lower (approximately 50%) than the one used by WiMAX devices. This can be seen in Table 15 where the Atmel radio chip consumes more energy than the Infineon radio chip since its supply voltage and current for each state are significantly higher.

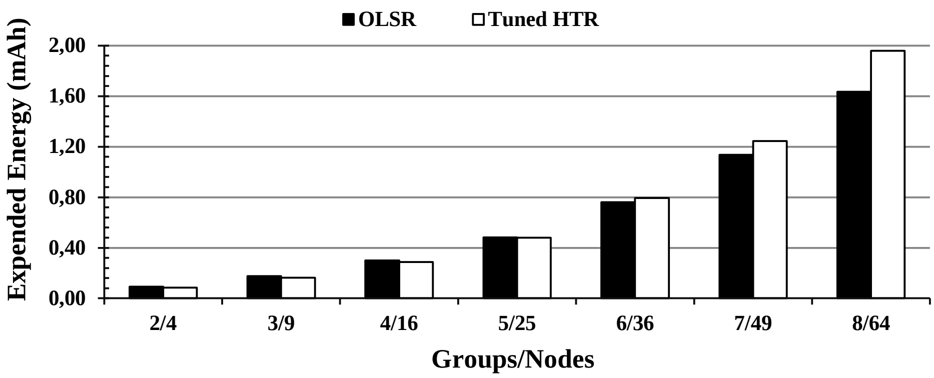


Figure 41: Comparing OLSR and tuned HTR Expended Energy when using WiMAX and Wi-Fi devices

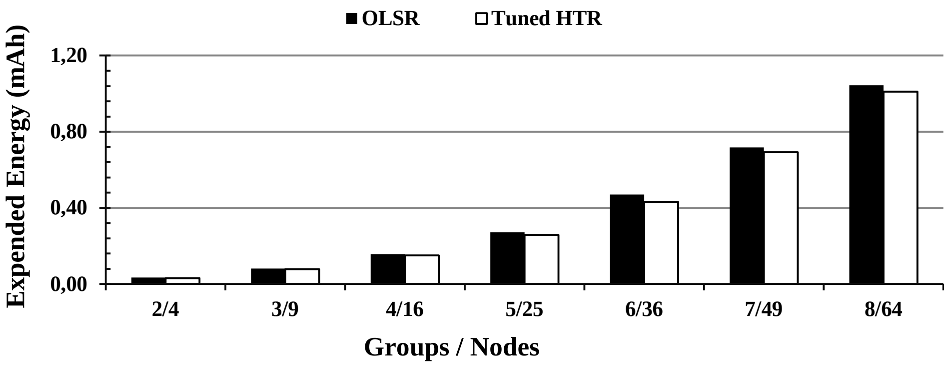


Figure 42: Comparing OLSR and tuned HTR Expended Energy when using LTE and Wi-Fi devices

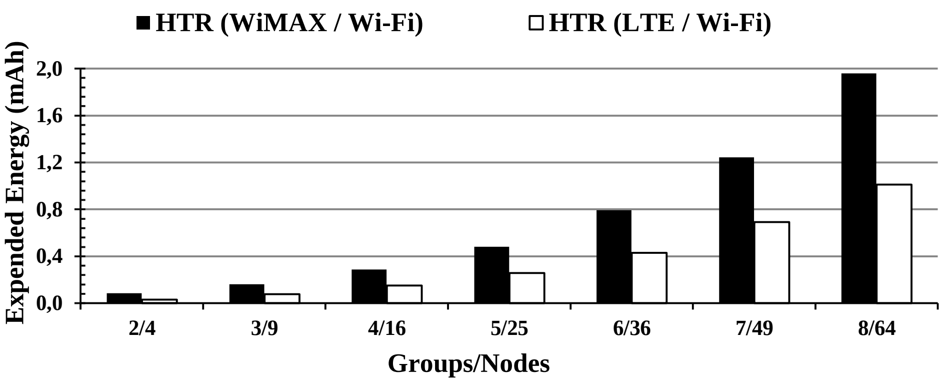


Figure 43: Comparing WiMAX and LTE using the Expended Energy metric and tuned HTR

5.2.2 RESULTS FOR SCENARIO B

This subsection shows the experimental results for Scenario B.

5.2.2.1 GOODPUT

This metric represents the average data delivery bitrate at a given destination and takes into account only the received data packets, disregarding the control messages. Figures Figure 44 and Figure 45 illustrate the comparison among Multipath HTR (i.e. with the proposed multipath extension), HTR (i.e. the original single-path protocol), and OLSR regarding goodput. Figure 44 shows the results for LTE and Wi-Fi nodes and Figure 45 for the combined use of WiMAX and Wi-Fi. One can note that this work proposal achieves very similar goodput results to OLSR and HTR at a 300 Kbps traffic rate. Nonetheless, in contrast, this work extension shows higher delivery rates at 600 Kbps, an increase of more than half (approximately 57%) and a twofold increase (225%) for LTE and Wi-Fi when comparing with OLSR and HTR, respectively. With the same bitrate, it shows an increase of 11% and 14% for WiMAX and Wi-Fi when compared with OLSR and HTR. At 1 Mbps, the gain reaches around 120% and 476% for LTE and Wi-Fi respectively and only 2% and 13.6% for WiMAX and Wi-Fi respectively.

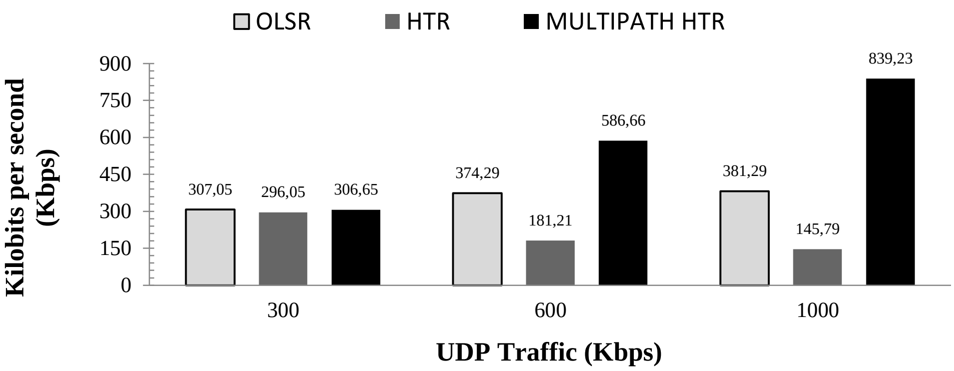


Figure 44: Goodput of LTE and Wi-Fi devices

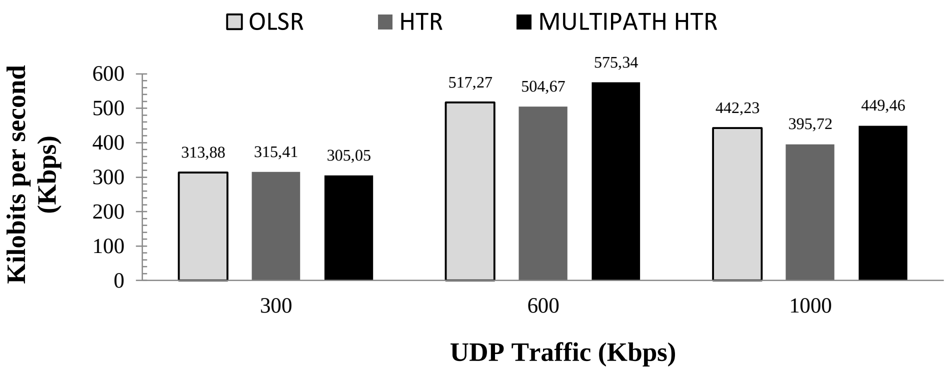


Figure 45: Goodput of WiMAX and Wi-Fi devices

5.2.2.2 EXPENDED ENERGY

Figure 46 shows the expended energy results for WiMAX and Wi-Fi and Figure 47 shows results for the LTE and Wi-Fi scenario. This metric indicates the energy consumption of the network and is obtained by summing the initial energy level of all network nodes and then subtracting from this value the sum of the resilient energy. In both cases, this work proposal shows minor impact on network energy consumption; it increased nearly 4.4% for the LTE and Wi-Fi scenario and 8.3% for the WiMAX and Wi-Fi scenario when compared to OLSR. In the case of multipath extension, the overhead is due to the additional header information included for routing loop control. Furthermore, this work's multipath routing strategy achieves a higher goodput, which also, understandably, impacts energy consumption results.

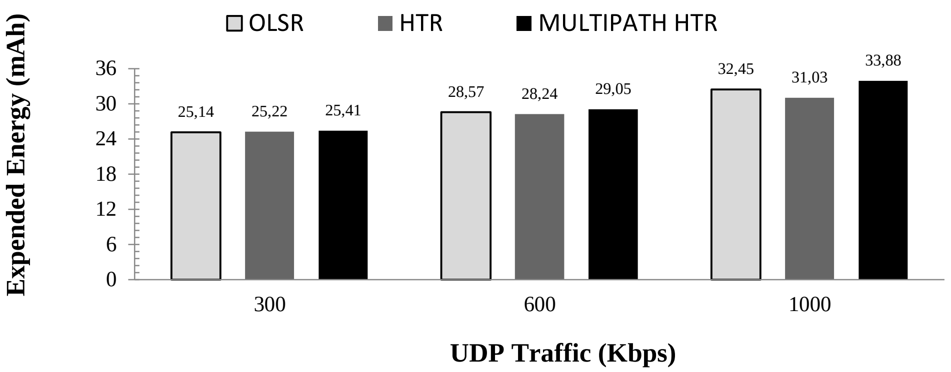


Figure 46: Expended Energy for the scenario using LTE and Wi-Fi devices

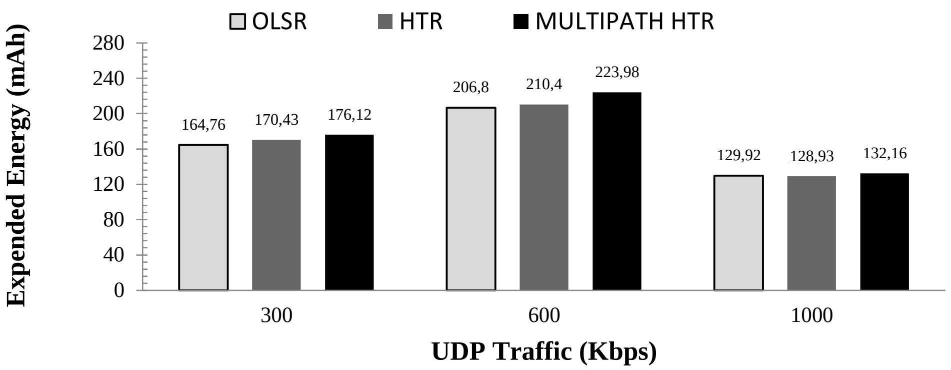


Figure 47: Expended Energy for the scenario using WiMAX and Wi-Fi devices

5.2.2.3 PACKET LOSS RATIO (PLR)

Figures Figure 48 and Figure 49 show the results of PLR. In Figure 48 (PLR with LTE and Wi-Fi), multipath HTR shows a lower packet loss ratio, in which OLSR had 3, 10 and 3.7 times more packet loses than the Multipath HTR at 300, 600 and 1000 Kbps rate, respectively. Furthermore, the HTR had a higher PLR when compared with its correspondent multipath version: about 7, 18 and 5 times higher, respectively, since routing loops occurred.

In Figure 49 (PLR with WiMAX and Wi-Fi), the Multipath HTR had the same PLR when compared with OLSR and 4 times less PLR than the single-path HTR at the 300 Kbps rate, and had nearly 4 times less PLR than both the OLSR and the single-path HTR at the 600 Kbps rate; however, at the 1000 Kbps rate, the Multipath HTR had almost the same PLR as the OSLR and single-path HTR.

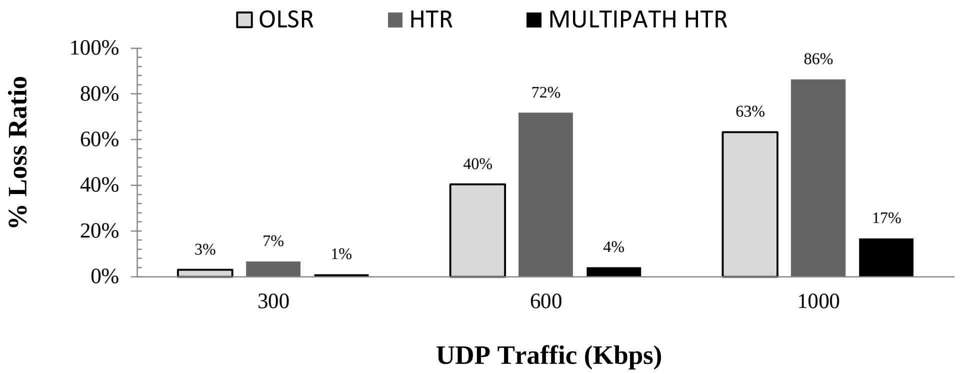


Figure 48: PLR for the scenario using LTE and Wi-Fi devices

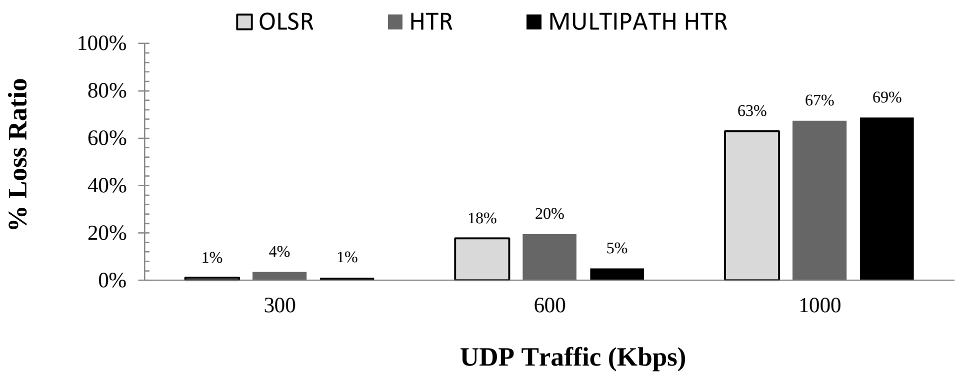


Figure 49: PLR for the scenario using WiMAX and Wi-Fi devices

5.2.2.4 Average End-to-end delay

Figures Figure 50 and Figure 51 show the results of average end-to-end delay. In Figure 50 (average end-to-end delay with LTE and Wi-Fi technologies), multipath HTR shows a lower delay, a decrease of 44.6%, 51% and 35.4% when compared with OLSR, and a decrease of 47.5%, 51% and 30.2% when compared with the HTR at 300, 600 and 1000 Kbps, respectively.

In Figure 51 (average end-to-end delay with WiMAX and Wi-Fi technologies), the average end-to-end delay was about the same at 300 Kbps, but at 600 and 1000 Kbps the

Multipath HTR shows a lower average end-to-end delay, with a decrease of 96% and 77.2% respectively when compared with OLSR and single-path HTR.

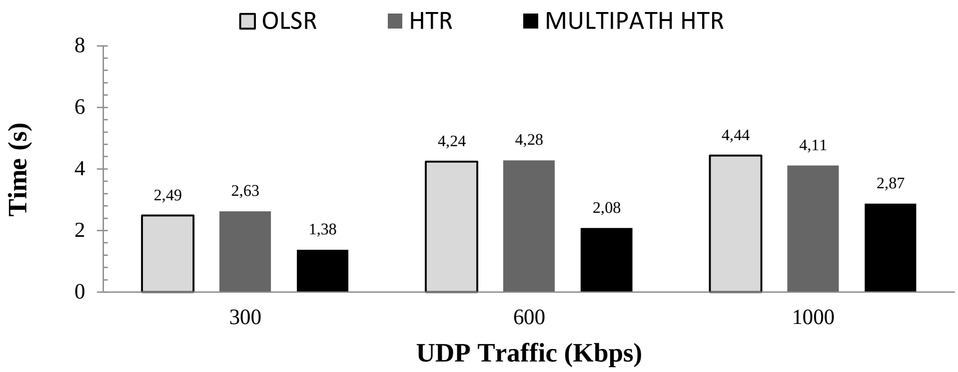


Figure 50: Average end-to-end delay for the scenario using LTE and Wi-Fi devices

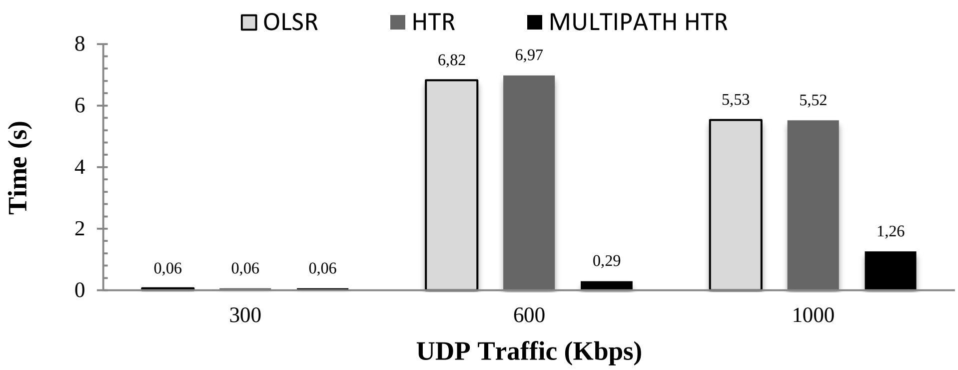


Figure 51: Average end-to-end delay for the scenario using WiMAX and Wi-Fi devices

5.2.3 DISCUSSION FOR SCENARIO B

Our results for scenario B show that, in terms of goodput, end-to-end average delay, and energy consumption, this work proposal improved the network performance with minimal impact on energy consumption. Moreover, the work proposed surpasses the data delivery rates and reduces significantly the end-to-end delay when compared with single-path HTR and the well-known OLSR. Additionally, one observes that the application of different heterogeneous technologies results in different network efficiencies. Moreover, our results indicate that the use of LTE rather than WiMAX leads to better network improvement, with minor end-to-end delay and packet losses, a better goodput, and improved energy consumption. Furthermore, it is clear that there was decrease of goodput and increase of packet loss ratio when using WiMAX technology at 1 Mbps. This behavior can be related to the maximum date rate expected for the technology when using the selected modulation scheme, coding rate, and propagation loss model. In [160], the author provides a prediction (2.55 Mbps for uplink using our parameters) of the WiMAX MAC layer goodput when varying the modulation scheme, packet length, coding rate, and number of users, and concludes that the goodput decreases for smaller packet sizes and for a larger number of users. Thus, for this reason, we conclude that the WiMAX base station was overloaded at 1 Mbps traffic rate.

5.3 CHAPTER SUMMARY

In this chapter, we presented the methodology used to evaluate the proposal and discussed the experimental results done. The methodology section explained the metrics and data collected, discussed the statistics methodology applied and the scenarios that compound the experiments, detailed the heterogeneous technologies and parameters, showed the propagation loss models used and their parameters and provided the configuration for the energy framework. The experimental results performed were divided in two parts: Scenario A and Scenario B. The first scenario defined the best HTR HELLO interval for the HTR protocol to use when varying the technology and network density. Furthermore, the scenario A evaluated the convergence time and expended energy of OLSR and Tuned HTR (HTR protocol when varying the HELLO interval). The Scenario B evaluated the performance of the Multipath HTR, which is the extension proposed in this work, OLSR and HTR baseline using as performance metrics the goodput, expended energy, packet loss ratio and average end-to-end delay.

CHAPTER 6 CONCLUSION

This work implemented the HTR, a protocol for connecting heterogeneous devices based on OLSR, using the ns-3 simulator and extended the energy framework of this simulator. These enhancements for the simulator include new device energy models that are used to track the energy consumption for devices using WiMAX and LTE technologies. Also, it studied the behavior of the HTR by modifying the HELLO interval and compared the results with the OLSR with default setup. The convergence time and the energy consumed were analyzed in this study and the HELLO interval with minimum convergence interval and least energy consumption was determined for each heterogeneous scenario and by varying the network density. Two 4G technologies were utilized, WiMAX and LTE, together with Wi-Fi and provide the heterogeneity of the scenarios.

This work concludes that varying the HELLO refresh interval can improve the convergence time and reduce the energy consumption without major impact on network behavior and that the tuned HTR outperforms the OLSR with default configuration. Furthermore, the propagation loss model can affect the performance of the 4G technologies. Currently, we are evaluating the use of other propagation loss models such as Okumura-Hata, Rayleigh, Walfisch-Ikegami, ECC33, and so forth in order to evaluate the impact of these propagation models on a heterogeneous network. In [150], the author provides an update for propagation loss prediction models and provides coverage notes that will help us to determine the most promising models to use in our context.

In this work, we also addressed the problem of routing loops on HTR by proposing a multipath extension that offers several benefits such as load balancing, routing loop prevention, energy-conservation, low end-to-end delay, and congestion avoidance, among others. Additionally, this work performed a comparative analysis of our proposal with the single-path HTR (baseline) and the widely used OLSR protocol. Results show that the proposed solution provides a more responsible protocol that effectively improves network performance by increasing the data delivery rate and reducing the end-to-end delay without a major impact on network energy consumption.

6.1 CONTRIBUTIONS

Contributions of this work are the following: first, a simulated version of the HTR protocol was given, providing an integrated solution for heterogeneous ad hoc communication scenarios, currently mostly lacking from industry and academia attention. As a second contribution, network performance was effectively increased under the multipath extension. Third, this work investigated the impact of tuning routing parameters on the protocol convergence interval and energy consumption that, to the best of our knowledge, is not evaluated in other works. Further, we discussed the impact of the heterogeneity, comparing the results obtained while using Wi-Fi, WiMAX and LTE technologies. Also, two new energy models for computation of energy consumption of WiMAX and LTE were developed and they enhance the ns-3 energy system which, at the moment of writing of this document, has only support for Wi-Fi devices. Furthermore, we submitted two articles, both in review process at the moment of writing of this dissertation, for two conferences. Finally, to the best of our knowledge, although a large number of methodologies to compare ad hoc network protocols have been published, none of them provides a methodology to compare these when using heterogeneous technologies; thus, our evaluation methodology sets a direction for the analysis of heterogeneous wireless MANETs and their protocols.

6.2 PROBLEMS ENCOUNTERED

During the development of the HTR protocol using the ns-3 simulator, we encountered some problems that are the following: first, there is a problem when we tried to setup base-stations from distinct 4G technologies such as WiMAX and LTE within the same experiment, we speculate that these stations are interfering with each other and we did not find a solution for this problem. Second, we were not able to setup an experiment with more than one WiMAX base-station, but this problem was not encountered for the LTE technology. Furthermore, the simulation took so much time to execute when the numbers of 4G nodes are greater than 12 and when the number of Wi-Fi devices was greater than 500.

6.3 FUTURE WORKS

For future work we are considering implementing an analytic model to compute the optimal quantity of distinct paths necessary for the multipath scheme. We would also be interested in including QoS requirements, improving routing context-awareness based on human roles or other context information, and finally, adding security features.

REFERENCES

[1] A. Akan and C. Edemen, “Path to 4G wireless networks,” in *2010 IEEE 21st International Symposium on Personal, Indoor and Mobile Radio Communications Workshops*, 2010, pp. 405–407 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5670404’ escapeXml='false'/>. [Accessed: 16-Mar-2013]

[2] Morgan Stanley, “The Mobile Internet Report,” 2009 [Online]. Available: http://www.mobitechintl.com/pdf/MS_MI_Report_Setup_12.09.pdf. [Accessed: 08-May-2013]

[3] M. Meeker and L. Wu, “2012 Internet Trends (Update) - Kleiner Perkins Caufield Byers,” 2012 [Online]. Available: http://www.kpcb.com/insights/2012-internet-trends-update. [Accessed: 08-May-2013]

[4] China Internet Network, “China Internet Development Statistics Report,” 2012 [Online]. Available: http://www.cnnic.cn/research/bgxz/tjbg/201207/P020120719489935146937.pdf

[5] StatCounter Global Stats, “Mobile vs Desktop Internet Traffic Report from December 2008 to May 2013.,” 2013 [Online]. Available: http://gs.statcounter.com/#mobile_vs_desktop-IN-monthly-200812-201305

[6] J. Li, S. Luo, S. Das, T. McAuley, M. Patel, A. Staikos, and M. Gerla, “An Integrated Framework for Seamless Soft Handoff in Ad Hoc Networks,” in *MILCOM 2006*, 2006, pp. 1–7 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4086867’ escapeXml='false'/>. [Accessed: 02-Mar-2013]

[7] DARPA Internet Program, “RFC 791 - Internet Protocol Specification.” IETF, pp. 1–45, 1981 [Online]. Available: http://tools.ietf.org/html/rfc791

[8] S. Deering and R. Hinden, “RFC 2460 - Internet Protocol Version 6 (IPv6) Specification.” IETF, pp. 1–39, 1998 [Online]. Available: http://tools.ietf.org/html/rfc2460

[9] A. Boukerche, B. Turgut, N. Aydin, M.Z. Ahmad, L. Bölöni, and D. Turgut, “Routing

protocols in ad hoc networks: A survey," *Computer Networks*, vol. 55, no. 13, pp. 3032–3080, Sep. 2011 [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2011.05.010. [Accessed: 28-Feb-2013]

[10] E. Souto, R. Aschoff, J. Lima Junior, R. Melo, D. Sadok, and J. Kelner, "HTR: A framework for interconnecting wireless heterogeneous devices," in *Consumer Communications and Networking Conference (CCNC), 2012 IEEE*, 2012, pp. 645–649.

[11] Ericsson Research, "Ericsson Labs | Research, Concepts, Innovation," 2013. [Online]. Available: https://labs.ericsson.com/. [Accessed: 08-May-2013]

[12] Networking and Telecommunication Research Group, "Welcome to GPRT !," 2013. [Online]. Available: https://www.gprt.ufpe.br/. [Accessed: 08-May-2013]

[13] MCTI, "PORTAL DO MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO," 2013. [Online]. Available: http://www.mct.gov.br/. [Accessed: 08-May-2013]

[14] C.M.T. Calafate, R.G. Garcia, and P. Manzoni, "Optimizing the implementation of a MANET routing protocol in a heterogeneous environment," in *Proceedings of the Eighth IEEE Symposium on Computers and Communications. ISCC 2003*, 2003, pp. 217–222 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=1214125&contentType=Conference+Publications&queryText=optimizing+the+implementation+of+a+manet+routing+protocol+in+a+heterogeneous+environment. [Accessed: 21-Mar-2013]

[15] R. Heydon, *Bluetooth Low Energy: The Developer's Handbook*, 1st ed. Prentice Hall, 2012, p. 368 [Online]. Available: http://www.amazon.com/dp/013288836X

[16] IEEE Standard for Information technology, "IEEE Std 802.11 - Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," vol. 2012, no. March. IEEE Computer Society, pp. 1–2793, 2012 [Online]. Available: http://standards.ieee.org/getieee802/download/802.11-2012.pdf

[17] IEEE Standard for Information technology, "Supplement to IEEE Standard for Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements — Part 11: Amendment 5: Enhancements for Higher Throughput." IEEE Computer Society, pp.

1–536, 2009 [Online]. Available: http://standards.ieee.org/getieee802/download/802.11n-2009.pdf

[18] IEEE Standard for Information technology, "Supplement to IEEE Standard for Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements — Part 11: Higher-Speed Physical Layer Extension in the 2.4 GHz Band," vol. 1999. IEEE Computer Society, pp. 1–96, 1999 [Online]. Available: http://standards.ieee.org/getieee802/download/802.11b-1999.pdf

[19] IEEE Standard for Information technology, "Supplement to IEEE Standard for Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements — Part 11: Amendment 4: Further Higher Data Rate Extension in the 2.4 GHz," vol. 2003, no. June. IEEE Computer Society, pp. 1–78, 2003 [Online]. Available: http://standards.ieee.org/getieee802/download/802.11g-2003.pdf

[20] Qualcomm incorporated, "IEEE802.11ac: The Next Evolution of Wi-Fi TM Standards," 2012 [Online]. Available: http://www.qualcomm.com/media/documents/files/qualcomm-research-ieee802-11ac-the-next-evolution-of-wi-fi.pdf

[21] Nokia, "Device Details -- Nokia N810," 2013. [Online]. Available: http://www.developer.nokia.com/Devices/Device_specifications/N810/. [Accessed: 08-May-2013]

[22] SAMSUNG, "Nexus S from Google | Samsung Nexus," 2013. [Online]. Available: http://www.samsung.com/us/mobile/cell-phones/GT-I9020FSTTMB. [Accessed: 08-May-2013]

[23] T.R. Henderson, S. Roy, S. Floyd, and G.F. Riley, "ns-3 project goals," in *Proceeding from the 2006 workshop on ns-2: the IP network simulator - WNS2 '06*, 2006, pp. 1–13 [Online]. Available: http://dl.acm.org/citation.cfm?id=1190455.1190468. [Accessed: 01-Mar-2013]

[24] D. Astly, E. Dahlman, A. Furuskr, Y. Jading, M. Lindstrm, and S. Parkvall, "LTE: the evolution of mobile broadband - [LTE part II: 3GPP release 8]," *Communications Magazine IEEE*, vol. 47, no. 4, pp. 44–51, 2009.

[25] H. Holma and A. Toskala, *LTE for UMTS: OFDMA and SC-FDMA based radio access*. John Wiley & Sons Ltd., 2009, p. 433 [Online]. Available: http://books.google.com/books?id=AHr43Lh-roQC&pgis=1

[26] S. Fraser, "Examining the Design and Test Challenges of 3GPP LTE," *Telecommunications*, pp. 1–14, 2007 [Online]. Available: http://proquest.umi.com.library.capella.edu/pqdweb?did=1401043191&Fmt=7&clientId=62763&RQT=309&VName=PQD

[27] B. Bjerke, *LTE-advanced and the evolution of LTE deployments*, vol. 18, no. 5. IEEE, 2011, pp. 4–5 [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/17579048

[28] T. Clausen and P. Jacquet, "RFC 3626 - Optimized Link State Routing Protocol (OLSR)." IETF, pp. 1–75, 01-Oct-2003 [Online]. Available: http://www.ietf.org/rfc/rfc3626.txt. [Accessed: 28-Feb-2013]

[29] C. Liu and J. Kaiser, "A Survey of Mobile Ad Hoc network Routing Protocols," 2003 [Online]. Available: http://d-nb.info/1015472427/34

[30] M. Tarique, K.E. Tepe, S. Adibi, and S. Erfani, "Survey of multipath routing protocols for mobile ad hoc networks," *Journal of Network and Computer Applications*, vol. 32, no. 6, pp. 1125–1143, Nov. 2009 [Online]. Available: http://dx.doi.org/10.1016/j.jnca.2009.07.002. [Accessed: 02-Feb-2013]

[31] M. Abolhasan, T. Wysocki, and E. Dutkiewicz, "A review of routing protocols for mobile ad hoc networks," *Ad Hoc Networks*, vol. 2, no. 1, pp. 1–22, Jan. 2004 [Online]. Available: http://dx.doi.org/10.1016/S1570-8705(03)00043-X. [Accessed: 05-Mar-2013]

[32] ETSI, "Broadband Radio Access Networks (BRAN); HIgh PErformance Radio Local Area Network (HIPERLAN) Type 1; Functional specification." pp. 1–105, 1998 [Online]. Available: http://easy.intranet.gr/H1.pdf

[33] T. Clausen, C. Dearlove, P. Jacquet, and U. Herberg, "The Optimized Link State Routing Protocol version 2 (draft 19)." IETF, pp. 1–113, 2013 [Online]. Available: http://www.ietf.org/id/draft-ietf-manet-olsrv2-19.txt

[34] U. Herberg, T. Clausen, and R. Cole, "MANET network management and performance monitoring for NHDP and OLSRv2," in *2010 International Conference on Network*

and Service Management, 2010, pp. 290–293 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5691209&contentType=Conference+Publications&refinements=4280529814&queryText=olsrv2. [Accessed: 28-Mar-2013]

[35] T. Clausen and U. Herberg, "Router and Link Admittance Control in the Optimized Link State Routing Protocol Version 2 (OLSRv2)," in *2010 Fourth International Conference on Network and System Security*, 2010, pp. 40–46 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5636079&contentType=Conference+Publications&refinements=4280529814&queryText=olsrv2. [Accessed: 28-Mar-2013]

[36] T. Clausen and U. Herberg, "Vulnerability analysis of the optimized link state routing protocol version 2 (OLSRv2)," in *2010 IEEE International Conference on Wireless Communications, Networking and Information Security*, 2010, pp. 628–633 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5544732&contentType=Conference+Publications&refinements=4280529814&queryText=olsrv2. [Accessed: 28-Mar-2013]

[37] T. Clausen, C. Dearlove, and B. Adamson, "RFC 5148 - Jitters Considerations in Mobile Ad Hoc Networks (MANETs)." IETF, pp. 1–12, 2008 [Online]. Available: http://tools.ietf.org/html/rfc5148

[38] T. Clausen, C. Dearlove, J.W. Dean, and C. Adjih, "RFC 5444 - Generalized Mobile Ad Hoc Network (MANET) Packet/Message Format." IETF, pp. 1–59, 2009 [Online]. Available: http://tools.ietf.org/html/rfc5444

[39] T. Clausen and C. Dearlove, "RFC 5497 - Representing Multi-Value Time in Mobile Ad Hoc Networks (MANETs)." IETF, pp. 1–14, 2009 [Online]. Available: http://tools.ietf.org/html/rfc5497

[40] U. Herberg and T. Clausen, "RFC 6622 - Integrity Check Value and Timestamp TLV Definitions for Mobile Ad Hoc Networks (MANETs)." IETF, pp. 1–21, 2012 [Online]. Available: http://tools.ietf.org/html/rfc6622

[41] T. Clausen, C. Dearlove, and J.W. Dean, "RFC 6130 - Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)." IETF, pp. 1–88, 2011 [Online].

Available: http://tools.ietf.org/html/rfc6130

[42] IETF, "Document history for draft-ietf-manet-olsrv2," 2013. [Online]. Available: http://datatracker.ietf.org/doc/draft-ietf-manet-olsrv2/history/. [Accessed: 28-Mar-2013]

[43] J. Crowcroft, "Mobile Ad-hoc Intern-domain Networking," in *Annual Conference of ITA*, 2007.

[44] C. Chau, J. Crowcroft, K. Lee, and S.H.Y. Wong, "IDRM : Inter-Domain Routing Protocol for Mobile Ad Hoc Networks," *Computer*, no. 708, 2008.

[45] J. Crowcroft, S. Hand, R. Mortier, T. Roscoe, and A. Warfield, "Plutarch : An Argument for Network Pluralism," in *Proceeding FDNA '03 Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture*, 2003, pp. 258–266.

[46] S. Schmid, L. Eggert, M. Brunner, and J. Quittek, "Towards autonomous network domains," in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, 2005, vol. 4, pp. 2847–2852 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1498574. [Accessed: 28-Feb-2013]

[47] N. Boulicault, G. Chelius, and E. Fleury, "Ana4 : a 2 . 5 Framework for Deploying Real Multi-hop Ad hoc and Mesh Networks," *Science*, vol. X, pp. 1–24, 2006.

[48] V. Untz, M. Heusse, F. Rousseau, and A. Duda, "Lilith: an interconnection architecture based on label switching for spontaneous edge networks," in *The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004.*, 2004, pp. 146–151 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1331721. [Accessed: 28-Feb-2013]

[49] B. Jacinto, L. Vilaça, and J. Kelner, "3D routing: a protocol for emergency scenarios," *Proceedings of the 6th International Wireless Communications and Mobile Computing Conference*, pp. 519–523, 2010 [Online]. Available: http://doi.acm.org/10.1145/1815396.1815516. [Accessed: 22-Feb-2013]

[50] E. Rosen and A. Viswanathan, "RFC 3031 - Multiprotocol Label Switching Architecture." IETF, pp. 1–61, 2001 [Online]. Available:

http://tools.ietf.org/html/rfc3031

[51] K. Sethom, M. Sabeur, B. Jouaber, H. Afifi, and D. Zeghlache, “Distributed virtual network interfaces to support intra-PAN and PAN-to-infrastructure connectivity,” in *GLOBECOM ’05. IEEE Global Telecommunications Conference, 2005.*, 2005, vol. 6, p. 5 pp.–3558 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1578434. [Accessed: 09-May-2013]

[52] M.K. Marina and S.R. Das, “On-demand multipath distance vector routing in ad hoc networks,” in *Proceedings Ninth International Conference on Network Protocols. ICNP 2001*, 2001, pp. 14–23 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=992756. [Accessed: 13-Mar-2013]

[53] C. Perkins, E. Royer, and S. Das, “RFC 3561 - Ad hoc On-Demand Distance Vector (AODV) Routing.” IETF, pp. 1–37, 2003 [Online]. Available: http://tools.ietf.org/html/rfc3561

[54] R. Leung, E. Poon, and A.-L.C. Chan, “MP-DSR: a QoS-aware multi-path dynamic source routing protocol for wireless ad-hoc networks,” in *Proceedings LCN 2001. 26th Annual IEEE Conference on Local Computer Networks*, 2001, pp. 132–141 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=990778. [Accessed: 13-Mar-2013]

[55] D. Maltz, Y. Hu, and D. Johnson, “RFC 4728 - The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4.” IETF, pp. 1–107, 2007 [Online]. Available: http://tools.ietf.org/html/rfc4728. [Accessed: 07-Oct-2010]

[56] J. Yi, A. Adnane, S. David, and B. Parrein, “Multipath optimized link state routing for mobile ad hoc networks,” *Ad Hoc Networks*, vol. 9, no. 1, pp. 28–47, Jan. 2011 [Online]. Available: http://dx.doi.org/10.1016/j.adhoc.2010.04.007. [Accessed: 22-Feb-2013]

[57] S.-J. Lee and M. Gerla, “Split multipath routing with maximally disjoint paths in ad hoc networks,” in *ICC 2001. IEEE International Conference on Communications. Conference Record (Cat. No.01CH37240)*, 2001, vol. 10, pp. 3201–3205 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=937262. [Accessed:

13-Mar-2013]

[58] Y. Huang, S. Bhatti, and D. Parker, "Tuning OLSR," in *2006 IEEE 17th International Symposium on Personal, Indoor and Mobile Radio Communications*, 2006, pp. 1–5 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=4022354&contentType=Conference+Publications&searchField=Search_All&queryText=tuning+olsr. [Accessed: 21-Mar-2013]

[59] P. Suganthi and A. Tamilarasi, "Performance Of OLSR Routing Protocol Under Different Route Refresh Intervals In Ad Hoc Networks," *International Journal on Computer Science and Engineering*, vol. 3, no. 1, pp. 133–137, 2011 [Online]. Available: http://www.enggjournals.com/ijcse/doc/IJCSE11-03-01-095.pdf

[60] T. Kunz and R. Alhalimi, "Energy-efficient proactive routing in MANET: Energy metrics accuracy," *Ad Hoc Networks*, vol. 8, no. 7, pp. 755–766, Sep. 2010 [Online]. Available: http://dx.doi.org/10.1016/j.adhoc.2010.02.004. [Accessed: 21-Mar-2013]

[61] P.E. Villanueva-Pena, T. Kunz, and P. Dhakal, "Extending network knowledge: making OLSR a quality of service conductive protocol," in *Proceedings of the International Conference on Communications and Mobile Computing*, 2006, pp. 103–108.

[62] T. Kunz and R. Alhalimi, "Load-Balanced Routing in Wireless Networks: State Information Accuracy Using OLSR," in *Third IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob 2007)*, 2007, pp. 83–83 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4390877. [Accessed: 21-Mar-2013]

[63] C. Gomez, D. Garcia, and J. Paradells, "Improving Performance of a Real Ad Hoc Network by Tuning OLSR Parameters," in *10th IEEE Symposium on Computers and Communications (ISCC'05)*, 2005, pp. 16–21 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=1493701&contentType=Conference+Publications&searchField=Search_All&queryText=tuning+olsr. [Accessed: 21-Mar-2013]

[64] C. Gomez, D. Garcia, and J. Paradells, "An OLSR parameter based study of the performance of real ad-hoc network environments," *Wireless Conference 2005 - Next*

Generation Wireless and Mobile Communications and Services (European Wireless), 11th European, pp. 1–6, 2005 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5755320&contentType=Conference+Publications&queryText=an+olsr+parameter+based+study+of+the+performance+of+real+ad-hoc+network. [Accessed: 21-Mar-2013]

[65] J. Hosek, "Performance Analysis of MANET Routing Protocols OLSR and AODV," *ElektrorevueEng*, vol. 2, no. 3, pp. 22–27, 2011 [Online]. Available: http://www.elektrorevue.cz/en/download/performance-analysis-of-manet-routing-protocols-olsr-and-aodv/

[66] A.P. Renold, B. Venkatalakshmi, and S. VijayaKumar, "Tuning parameters of AODV routing protocol for improved performance in volcano monitoring," in *2011 3rd International Conference on Electronics Computer Technology*, 2011, pp. 373–376 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5941721&contentType=Conference+Publications&searchField=Search_All&queryText=tuning+parameters+of+aodv+routing+protocol+for+improved+performance+in+volcano+monitoring. [Accessed: 21-Mar-2013]

[67] R. Aschoff, E. Souto, J. Kelner, and D. Sadok, "(WO2011009177) NETWORK ADDRESS ALLOCATION METHOD," U.S. Patent PCT/BR2009/0002152011.

[68] P. Ji, Z. Ge, J. Kurose, and D. Towsley, "A Comparison of Hard-State and Soft-State Signaling Protocols," *IEEE/ACM Transactions on Networking*, vol. 15, no. 2, pp. 281–294, Apr. 2007 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=4154757&contentType=Journals+&+Magazines&searchField=Search_All&queryText=a+comparison+of+hard-state+and+soft-state+protocols. [Accessed: 22-Mar-2013]

[69] R.R. Aschoff, "Address Autoconfiguration for Mobile Ad-hoc Network based on Binary Number System," Universidade Federal de Pernambuco, 2010.

[70] IEEE Standard for Information technology, "754-2008 - IEEE Standard for Floating-Point Arithmetic," pp. 1–58, 2008 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=4610935&contentType=Standards&sortType=asc_p_Sequence&filter=AND(p_Publication_Number:4610933). [Accessed: 29-Mar-2013]

[71] P. Jacquet, P. Minet, P. Mühlethaler, and N. Rivierre, "Increasing Reliability in Cable-Free Radio LANs Low Level Forwarding in HIPERLAN," *Wireless Personal Communications*, vol. 4, no. 1, pp. 51–63, 1997 [Online]. Available: http://link.springer.com/article/10.1023/A:1008861218989. [Accessed: 01-Apr-2013]

[72] O. Liang, Y. Sekercioglu, and N. Mani, "A survey of multipoint relay based broadcast schemes in wireless ad hoc networks," *IEEE Communications Surveys & Tutorials*, vol. 8, no. 4, pp. 30–46, 2006 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4062838. [Accessed: 02-Apr-2013]

[73] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking - MobiCom '99*, 1999, pp. 151–162 [Online]. Available: http://dl.acm.org/citation.cfm?id=313451.313525. [Accessed: 02-Apr-2013]

[74] L. Viennot, "Complexity Results on Election of Multipoint Relays in Wireless Networks," INRIA RR-3835, 1998 [Online]. Available: http://hal.inria.fr/docs/00/07/30/97/PDF/RR-3584.pdf. [Accessed: 02-Apr-2013]

[75] P. Jacquet, A. Laouiti, P. Minet, and L. Viennot, "Performance analysis of OLSR Multipoint Relay flooding in two ad hoc wireless network models." INRIA, pp. 1–26, 2001 [Online]. Available: http://hal.archives-ouvertes.fr/docs/00/07/23/27/PDF/RR-4260.pdf

[76] P. Jacquet, A. Laouiti, P. Minet, and L. Viennot, "Performance of Multipoint Relaying in Ad Hoc Mobile Routing Protocols," *Proceeding NETWORKING '02 Proceedings of the Second International IFIP-TC6 Networking Conference on Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; and Mobile and Wireless Communications*, pp. 387–398, May 2002 [Online]. Available: http://dl.acm.org/citation.cfm?id=647077.714350. [Accessed: 02-Apr-2013]

[77] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint relaying for flooding broadcast messages in mobile wireless networks," in *Proceedings of the 35th Annual Hawaii International Conference on System Sciences*, 2002, pp. 3866–3875 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=994521. [Accessed:

02-Apr-2013]

[78] E.W. Dijkstra, “A note on two problems in connexion with graphs,” *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, Dec. 1959 [Online]. Available: http://link.springer.com/10.1007/BF01386390. [Accessed: 23-Mar-2013]

[79] L. Tassiulas, “Energy conserving routing in wireless ad-hoc networks,” in *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*, 2000, vol. 1, pp. 22–31 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=832170&contentType=Conference+Publications&queryText=Energy+conserving+routing+in+wireless+ad+hoc+networks. [Accessed: 29-Mar-2013]

[80] J. Yi, E. Cizeron, S. Hamma, and B. Parrein, “NET 08-2 - Simulation and Performance Analysis of MP-OLSR for Mobile Ad Hoc Networks,” *2008 IEEE Wireless Communications and Networking Conference*, pp. 2235–2240, Mar. 2008 [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4489426

[81] S. Mueller, R. Tsang, and D. Ghosal, “Multipath routing in mobile ad hoc networks: Issues and challenges,” *Performance Tools and Applications to …*, vol. 2965, pp. 209–234, 2004 [Online]. Available: http://www.springerlink.com/index/2HNXE3Q8F0Q4TCTW.pdf. [Accessed: 22-Feb-2013]

[82] D. Kachan, “Integration of NS-3 with MATLAB/Simulink,” Lulea University of Technology, 2010 [Online]. Available: http://epubl.ltu.se/1653-0187/2010/062/LTU-PB-EX-10062-SE.pdf. [Accessed: 22-Feb-2013]

[83] G.S. Fishman, *Principles of Discrete Event Simulation*. John Wiley & Sons Inc, 1979, p. 534 [Online]. Available: http://www.amazon.com/dp/0471043958

[84] E. Weingartner, H. vom Lehn, and K. Wehrle, “A Performance Comparison of Recent Network Simulators,” in *2009 IEEE International Conference on Communications*, 2009, pp. 1–5 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5198657’ escapeXml='false'/>. [Accessed: 31-Jan-2013]

[85] D. Pal, “A Comparative Analysis of Modern Day Network Simulators,” *Advances in Intelligent Systems and Computing*, vol. 167, pp. 489–498, 2012 [Online]. Available: http://www.springerlink.com/index/10.1007/978-3-642-30111-7. [Accessed: 04-Mar-2013]

[86] T. Issariyakul and E. Hossain, *Introduction to network simulator NS2*, 2nd ed. 20. Springer, 2011, p. 536 [Online]. Available: http://www.amazon.com/dp/1461414059. [Accessed: 22-Feb-2013]

[87] Y. Xue, H.S. Lee, M. Yang, P. Kumarawadu, H.H. Ghenniwa, and W. Shen, “Performance Evaluation of NS-2 Simulator for Wireless Sensor Networks,” in *2007 Canadian Conference on Electrical and Computer Engineering*, 2007, pp. 1372–1375 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4233004’ escapeXml='false'/>. [Accessed: 02-Mar-2013]

[88] B. Stroustrup, *The C++ Programming Language: Special Edition*. Addison-Wesley Professional, 2000, p. 1030 [Online]. Available: http://www.amazon.com/The-Programming-Language-Special-Edition/dp/0201700735. [Accessed: 22-Feb-2013]

[89] otcl-tclcl-developers, “OTcl.” [Online]. Available: http://otcl-tclcl.sourceforge.net/otcl/. [Accessed: 02-Mar-2013]

[90] X. Xian, W. Shi, and H. Huang, “Comparison of OMNET++ and other simulator for WSN simulation,” in *2008 3rd IEEE Conference on Industrial Electronics and Applications*, 2008, pp. 1439–1443 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4582757. [Accessed: 04-Mar-2013]

[91] R. Barr, Z.J. Haas, and R. van Renesse, “JiST: an efficient approach to simulation using virtual machines: Research Articles,” *Software—Practice & Experience*, vol. 35, no. 6, pp. 539–576, May 2005 [Online]. Available: http://dl.acm.org/citation.cfm?id=1060168.1060170. [Accessed: 04-Mar-2013]

[92] H. Schildt, *Java The Complete Reference, 8th Edition*, 8th ed. McGraw-Hill Osborne Media, 2011, p. 1152.

[93] A. Bahouth, S. Crites, N. Matloff, and T. Williamson, “Revisiting the Issue of Performance Enhancement of Discrete Event Simulation Software,” in *40th Annual*

Simulation Symposium (ANSS'07), 2007, pp. 114–122 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=4127209&contentType=Conference+Publications&searchField=Search_All&queryText=simpy. [Accessed: 04-Mar-2013]

[94] M. Lutz, *Programming Python*, Fourth Edi. O'Reilly Media, 2011, p. 1632 [Online]. Available: http://www.amazon.com/dp/0596158106

[95] G.F. Riley, "Large-scale network simulations with GTNetS," in *Proceedings of the 2003 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.03EX693)*, 2003, vol. 1, pp. 676–684 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1261483' escapeXml='false'/>. [Accessed: 02-Mar-2013]

[96] Nsnam, "ns-3 Tutorial (Release ns-3.15)," 2012. [Online]. Available: http://www.nsnam.org/docs/release/3.15/tutorial/ns-3-tutorial.pdf. [Accessed: 22-Feb-2013]

[97] R.W. Smith, *LPIC-1: Linux Professional Institute Certification Study Guide:*, 3rd ed. Sybex, 2013, p. 648 [Online]. Available: http://www.amazon.com/dp/1118495632

[98] M. Russinovich, D.A. Solomon, and A. Ionescu, *Windows® Internals: Including Windows Server 2008 and Windows Vista, Fifth Edition*, 5th ed. Microsoft Press, 2009, p. 1232 [Online]. Available: http://www.amazon.com/dp/0735625301

[99] E. Bott, C. Siechert, and C. Stinson, *Windows® 7 Inside Out*, Pap/Cdr. Microsoft Press, 2011, p. 1000 [Online]. Available: http://www.amazon.com/dp/0735626650. [Accessed: 22-Feb-2013]

[100] T. Northrup, *Windows® 8 Inside Out*. Microsoft Press, 2012, p. 752 [Online]. Available: http://www.amazon.com/dp/0735663815

[101] A. Laurent, *Understanding open source and free software licensing*, First Edit. O'Reilly Media, 2004, p. 224 [Online]. Available: http://www.amazon.com/dp/0596005814. [Accessed: 23-Feb-2013]

[102] Nsnam, "ns-3-dev: RELEASE_NOTES@d5b9559ce074." pp. 1–39, 2013 [Online]. Available: http://code.nsnam.org/ns-3-dev/file/d5b9559ce074/RELEASE_NOTES. [Accessed: 23-Feb-2013]

[103] H. Wu, S. Nabar, and R. Poovendran, “An Energy Framework for the Network Simulator 3 (ns-3),” *Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques*, vol. 3, pp. 222–230, 2011 [Online]. Available: http://eudl.eu/doi/10.4108/icst.simutools.2011.245534

[104] K. Lahiri and S. Dey, “Battery-driven system design: A new frontier in low power design,” *Design Automation Conference, 2002. Proceedings of ASP-DAC 2002. 7th Asia and South Pacific and the 15th International Conference on VLSI Design. Proceedings.*, pp. 261–267, 2002 [Online]. Available: http://dl.acm.org/citation.cfm?id=835514. [Accessed: 22-Feb-2013]

[105] M. Spohn and S. Sausen, “Simulation of blind flooding over wireless sensor networks based on a realistic battery model,” *Networking, 2008. ICN …*, pp. 545–550, 2008 [Online]. Available: http://dx.doi.org/10.1109/ICN.2008.100. [Accessed: 22-Feb-2013]

[106] D.N. Rakhmatov and S.B.K. Vrudhula, “An analytical high-level battery model for use in energy management of portable electronic systems,” *IEEE/ACM International Conference on Computer Aided Design (ICCAD) 2001*, pp. 488–493, Nov. 2001 [Online]. Available: http://dl.acm.org/citation.cfm?id=603095.603196. [Accessed: 22-Feb-2013]

[107] D. Rakhmatov, S. Vrudhula, and D.A. Wallach, “Battery lifetime prediction for energy-aware computing,” in *Proceedings of the 2002 international symposium on Low power electronics and design - ISLPED ’02*, 2002, pp. 154–159 [Online]. Available: http://dl.acm.org/citation.cfm?id=566408.566449. [Accessed: 22-Feb-2013]

[108] Nsnam, “Li-Ion model fitting - Nsnam,” 2010. [Online]. Available: http://www.nsnam.org/wiki/index.php/Li-Ion_model_fitting. [Accessed: 22-Feb-2013]

[109] C.M. Shepherd, *THEORETICAL DESIGN OF PRIMARY AND SECONDARY CELLS. PART 3. BATTERY DISCHARGE EQUATION*. PN, 1963, pp. 1–15 [Online]. Available: http://www.amazon.com/THEORETICAL-PRIMARY-SECONDARY-DISCHARGE-EQUATION/dp/B009XLW0QE. [Accessed: 22-Feb-2013]

[110] O. Tremblay, L.-A. Dessaint, and A.-I. Dekkiche, “A Generic Battery Model for the Dynamic Simulation of Hybrid Electric Vehicles,” in *2007 IEEE Vehicle Power and*

Propulsion Conference, 2007, pp. 284–289 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4544139' escapeXml='false'/>. [Accessed: 31-Jan-2013]

[111] Panasonic, "Panasonic CGR18650DA (LITHIUM ION BATTERIES: INDIVIDUAL DATA SHEET)," 2007 [Online]. Available: http://www.panasonic.com/industrial/includes/pdf/Panasonic_LiIon_CGR18650DA.pdf. [Accessed: 22-Feb-2013]

[112] IEEE Standard for Information technology, "Supplement to IEEE Standard for Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements — Part 11 : Higher-Speed Physical Layer Extension in the 2.4 GHz Band," vol. 1999, no. June. IEEE Computer Society, pp. 1–91, 1999 [Online]. Available: http://standards.ieee.org/getieee802/download/802.11a-1999.pdf

[113] ATHEROS COMMUNICATIONS, "Power Consumption and Energy Efficiency Comparisons of WLAN Products," 2004 [Online]. Available: http://www.qca.qualcomm.com/media/resource/resource_15_file2.pdf. [Accessed: 22-Feb-2013]

[114] M. Ergen and P. Varaiya, "Decomposition of Energy Consumption in IEEE 802.11," in *2007 IEEE International Conference on Communications*, 2007, pp. 403–408 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4288744' escapeXml='false'/>. [Accessed: 22-Feb-2013]

[115] Texas Instruments, "CC2420 2.4 GHz IEEE 802.15. 4/ZigBee-ready RF Transceiver," 2004 [Online]. Available: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:CC2420+(2.4+GHz+IEEE+802.15.4+/+ZigBee-ready+RF+Transceiver)#0. [Accessed: 22-Feb-2013]

[116] Nsnam, "ns-3 Manual (Release ns-3.15)," 2012. [Online]. Available: http://www.nsnam.org/docs/release/3.15/manual/ns-3-manual.pdf. [Accessed: 22-Feb-2013]

[117] Atmel, "Atmel AT86RF535B (3.5 Ghz WiMAX Transceiver)," 2007 [Online]. Available: http://www.gaw.ru/pdf/Atmel/wimax/AT86RF535B.pdf. [Accessed: 22-Feb-2013]

[118] Infineon, “Infineon BGA777L7 Data Sheet,” 2009 [Online]. Available: http://www.infineon.com/dgdl/BGA777L7.pdf?folderId=db3a30431f848401011fcbf2ab4c04c4&fileId=db3a3043284aacd80128b4a2b63100fb. [Accessed: 22-Feb-2013]

[119] Nsnam, “PyViz - Nsnam,” 2011. [Online]. Available: http://www.nsnam.org/wiki/index.php/PyViz. [Accessed: 27-Feb-2013]

[120] Gnome Project, “The GTK+ Project,” 2012. [Online]. Available: http://www.gtk.org/. [Accessed: 27-Feb-2013]

[121] Gnome Project and PyGtk Team, “PyGTK,” 2010. [Online]. Available: http://www.pygtk.org/. [Accessed: 27-Feb-2013]

[122] Gnome Project, “GooCanvas - GNOME Live!,” 2011. [Online]. Available: https://live.gnome.org/GooCanvas. [Accessed: 27-Feb-2013]

[123] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, 1st ed. Addison-Wesley Professional, 1994, p. 416 [Online]. Available: http://www.amazon.com/dp/0201633612

[124] E.M. Royer, “A review of current routing protocols for ad hoc mobile wireless networks,” *IEEE Personal Communications*, vol. 6, no. 2, pp. 46–55, Apr. 1999 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=760423&contentType=Journals+&+Magazines&queryText=a+review+of+current+routing+protocols+for+ad+hoc+mobile+wireless+network. [Accessed: 28-Mar-2013]

[125] J. Broch, D.A. Maltz, D.B. Johnson, Y.-C. Hu, and J. Jetcheva, “A performance comparison of multi-hop wireless ad hoc network routing protocols,” in *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking - MobiCom ’98*, 1998, pp. 85–97 [Online]. Available: http://dl.acm.org/citation.cfm?id=288235.288256. [Accessed: 18-Mar-2013]

[126] S.R. Das, C.E. Perkins, and E.M. Royer, “Performance comparison of two on-demand routing protocols for ad hoc networks,” in *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*, 2000, vol. 1, pp. 3–12 [Online]. Available:

http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=832168&contentType=Conference+Publications&queryText=Performance+comparison+of+two+on-demand+routing+protocols+for+ad+hoc+networks. [Accessed: 04-Mar-2013]

[127] S. Corson and J. Macker, "RFC 2501 - Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations." IETF, pp. 1–12, 1999 [Online]. Available: https://tools.ietf.org/html/rfc2501

[128] G. Carneiro, P. Fortuna, and M. Ricardo, "FlowMonitor - a network monitoring framework for the Network Simulator 3 (NS-3)," in *Proceedings of the 4th International ICST Conference on Performance Evaluation Methodologies and Tools*, 2009, pp. 1–10 [Online]. Available: http://dl.acm.org/citation.cfm?id=1698822.1698824. [Accessed: 03-Apr-2013]

[129] D. Obradovic, "Real-time model and convergence time of BGP," in *Proceedings.Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, 2002, vol. 2, pp. 893–901 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=1019336&contentType=Conference+Publications&sortType=desc_p_Citation_Count&searchField=Search_All&queryText=routing+.QT.convergence+time.QT. [Accessed: 03-Apr-2013]

[130] A. Gyorgy and T. Borsos, "Estimates on the packet loss ratio via queue tail probabilities," in *GLOBECOM'01. IEEE Global Telecommunications Conference (Cat. No.01CH37270)*, 2001, vol. 4, pp. 2407–2411 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=966209&contentType=Conference+Publications&searchField=Search_All&queryText=.QT.packet+loss+ratio.QT. [Accessed: 03-Apr-2013]

[131] P. Li, Y. Fang, and J. Li, "Throughput, Delay, and Mobility in Wireless Ad Hoc Networks," in *2010 Proceedings IEEE INFOCOM*, 2010, pp. 1–9 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5461902&contentType=Conference+Publications&sortType=desc_p_Citation_Count&searchField=Search_All&queryText=.QT.average+end-to-end+delay.QT. [Accessed: 03-Apr-2013]

[132] D.R. Anderson, D.J. Sweeney, and T.A. Williams, *Statistics for Business and Economics*, 11th ed. South-Western College Pub, 2010, p. 1080 [Online]. Available: http://www.amazon.com/dp/0324783248

[133] H. Fischer, *A History of the Central Limit Theorem: From Classical to Modern Probability Theory*, 2011th ed. Springer, 2010, p. 418 [Online]. Available: http://www.amazon.com/dp/0387878564

[134] S.S. Shapiro and M.B. Wilk, “An Analysis of Variance Test for Normality (Complete Samples),” *Biometrika*, vol. 52, pp. 591–611, 1965.

[135] J.P. Royston, “An Extension of Shapiro and Wilk’s W Test for Normality to Large Samples,” *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 31, no. 2, pp. 115–124, 1982 [Online]. Available: http://www.jstor.org/stable/2347973

[136] A. Sen and M. Srivastava, *Regression Analysis: Theory, Methods, and Applications*, Softcover . Springer, 2011, p. 364 [Online]. Available: http://www.amazon.com/dp/1461287898

[137] M.D. Smucker, J. Allan, and B. Carterette, “A comparison of statistical significance tests for information retrieval evaluation,” in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management - CIKM ’07*, 2007, pp. 623–632 [Online]. Available: http://dl.acm.org/citation.cfm?id=1321440.1321528. [Accessed: 25-Mar-2013]

[138] S. Voulgaropoulou, G. Spanos, and L. Angelis, “Analyzing Measurements of the R Statistical Open Source Software,” in *2012 35th Annual IEEE Software Engineering Workshop*, 2012, pp. 1–10 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6479797&contentType=Conference+Publications&sortType=asc_p_Sequence&filter=AND(p_IS_Number:6479789). [Accessed: 03-Apr-2013]

[139] J.-S. Lee, Y.-W. Su, and C.-C. Shen, “A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee, and Wi-Fi,” in *IECON 2007 - 33rd Annual Conference of the IEEE Industrial Electronics Society*, 2007, pp. 46–51 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4460126. [Accessed: 03-Apr-2013]

[140] R. Pyndiah, A. Picart, and A. Glavieux, “Performance of block turbo coded 16-QAM and 64-QAM modulations,” in *Proceedings of GLOBECOM ’95*, 1995, vol. 2, pp. 1039–1043 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=502563&contentTyp

e=Conference+Publications&searchField=Search_All&queryText=64+quadrature+amplitude+modulation. [Accessed: 03-Apr-2013]

[141] V. Tarokh, H. Jafarkhani, and A.R. Calderbank, "Space-time block codes from orthogonal designs," *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1456–1467, Jul. 1999 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=771146&contentType=Journals+&+Magazines&sortType=desc_p_Citation_Count&searchField=Search_All&queryText=quadrature+amplitute+modulation+64-qam. [Accessed: 08-Mar-2013]

[142] G. Ungerboeck, "Channel coding with multilevel/phase signals," *IEEE Transactions on Information Theory*, vol. 28, no. 1, pp. 55–67, Jan. 1982 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=1056454&contentType=Journals+&+Magazines&sortType=desc_p_Citation_Count&searchField=Search_All&queryText=quadrature+amplitute+modulation+64-qam. [Accessed: 03-Apr-2013]

[143] L. Cimini, "Analysis and Simulation of a Digital Mobile Channel Using Orthogonal Frequency Division Multiplexing," *IEEE Transactions on Communications*, vol. 33, no. 7, pp. 665–675, Jul. 1985 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=1096357&contentType=Journals+&+Magazines&sortType=desc_p_Citation_Count&searchField=Search_All&queryText=ofdm. [Accessed: 03-Apr-2013]

[144] G. Bianchi, F. Formisano, and D. Giustiniano, "802.11b/g Link Level Measurements for an Outdoor Wireless Campus Network," in *2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks(WoWMoM'06)*, 2006, pp. 525–530 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1648508. [Accessed: 03-Apr-2013]

[145] V. Armoogum, K.M.S. Soyjaudah, N. Mohamudally, and T. Fogarty, "Comparative Study of Path Loss Using Existing Models for Digital Television Broadcasting for Summer Season in the North of Mauritius," in *The Third Advanced International Conference on Telecommunications (AICT'07)*, 2007, pp. 34–34 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=4215255&contentType=Conference+Publications&searchField=Search_All&queryText=comparative+study+of+path+loss+using+existing+models. [Accessed: 04-Apr-2013]

[146] V.S. Abhayawardhana, I.J. Wassell, D. Crosby, M.P. Sellars, and M.G. Brown, "Comparison of Empirical Propagation Path Loss Models for Fixed Wireless Access Systems," in *2005 IEEE 61st Vehicular Technology Conference*, vol. 1, pp. 73–77 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=1543252&contentType=Conference+Publications&searchField=Search_All&queryText=comparison+of+empirical+propagation+path+loss+models+for+fixed. [Accessed: 04-Apr-2013]

[147] S. Al-Rashdy, "Capacity and cell range estimation of mobile WiMAX," in *4th IET International Conference on Wireless, Mobile & Multimedia Networks (ICWMMN 2011)*, 2011, pp. 30–34 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6197819&contentType=Conference+Publications&searchField=Search_All&queryText=capacity+and+cell+range+estimation+of+mobile+wimax. [Accessed: 04-Apr-2013]

[148] M. Stoffers and G. Riley, "Comparing the ns-3 Propagation Models," in *2012 IEEE 20th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, 2012, pp. 61–67 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?reload=true&arnumber=6298165. [Accessed: 04-Mar-2013]

[149] T.S. Rappaport, *Wireless Communications: Principles and Practice*, 2nd ed. Prentice Hall, 2002, p. 736 [Online]. Available: http://www.amazon.com/dp/0130422320

[150] C. Phillips, D. Sicker, and D. Grunwald, "A Survey of Wireless Path Loss Prediction and Coverage Mapping Methods," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 1, pp. 255–270, 2013 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6165627. [Accessed: 04-Mar-2013]

[151] J.S. Seybold, *Introduction to RF Propagation*, 1st ed. Wiley-Interscience, 2005, p. 352 [Online]. Available: http://www.amazon.com/dp/0471655961

[152] D. Cox, "Delay Doppler characteristics of multipath propagation at 910 MHz in a suburban mobile radio environment," *IEEE Transactions on Antennas and Propagation*, vol. 20, no. 5, pp. 625–635, Sep. 1972 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=1140277&contentType=Journals+&+Magazines&sortType=desc_p_Citation_Count&searchField=Search_A

ll&queryText=doppler+spreading. [Accessed: 04-Apr-2013]

[153] M.S. Ab-Rahman, M.F. Ibrahim, and A.A.A. Rahni, “Thermal Noise Effect in FTTH Communication Systems,” in *2008 Fourth Advanced International Conference on Telecommunications*, 2008, pp. 364–370 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=4545557&contentType=Conference+Publications&queryText=thermal+noise. [Accessed: 04-Apr-2013]

[154] H.T. Friis, “A Note on a Simple Transmission Formula,” *Proceedings of the IRE*, vol. 34, no. 5, pp. 254–256, May 1946 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1697062. [Accessed: 21-Feb-2013]

[155] Centre Tecnològic de Telecomunicacions de Catalunya (CTTC), “The LTE-EPC Network Simulator (LENA) project,” 2013. [Online]. Available: http://iptechwiki.cttc.es/LTE-EPC_Network_Simulator_(LENA)

[156] M. Hata, “Empirical formula for propagation loss in land mobile radio services,” *IEEE Transactions on Vehicular Technology*, vol. 29, no. 3, pp. 317–325, Aug. 1980 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1622772. [Accessed: 19-Feb-2013]

[157] M. Alshami, T. Arslan, J. Thompson, and A. Erdogan, “Evaluation of Path Loss Models at WiMAX Cell-Edge,” in *2011 4th IFIP International Conference on New Technologies, Mobility and Security*, 2011, pp. 1–5 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5721070&contentType=Conference+Publications&searchField%3DSearch_All%26queryText%3Devaluation+of+path+loss+models+at+wimax+cell-edge. [Accessed: 04-Apr-2013]

[158] M. Alshami, T. Arslan, J. Thompson, and A.T. Erdogan, “Frequency analysis of path loss models on WIMAX,” in *2011 3rd Computer Science and Electronic Engineering Conference (CEEC)*, 2011, pp. 1–6 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5995815&contentType=Conference+Publications&searchField=Search_All&queryText=frequency+analysis+of+path+loss+models+on+wimax. [Accessed: 04-Apr-2013]

[159] C.E. Perkins and P. Bhagwat, “Highly dynamic Destination-Sequenced Distance-Vector

routing (DSDV) for mobile computers," in *Proceedings of the conference on Communications architectures, protocols and applications - SIGCOMM '94*, 1994, vol. 24, no. 4, pp. 234–244 [Online]. Available: http://dl.acm.org/citation.cfm?id=190314.190336. [Accessed: 28-Feb-2013]

[160] D. Pareit, V. Petrov, B. Lannoo, E. Tanghe, W. Joseph, I. Moerman, P. Demeester, and L. Martens, "A Throughput Analysis at the MAC Layer of Mobile WiMAX," in *2010 IEEE Wireless Communication and Networking Conference*, 2010, pp. 1–6 [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5506379&contentType=Conference+Publications&searchField=Search_All&queryText=wimax+data+rate. [Accessed: 20-Mar-2013]

[161] Nsnam, "Python bindings - Nsnam," 2013. [Online]. Available: http://www.nsnam.org/wiki/index.php/Python_bindings. [Accessed: 23-Feb-2013]

[162] G. Carneiro, "pybindgen - Python Bindings Generator - Google Project Hosting." [Online]. Available: http://code.google.com/p/pybindgen/. [Accessed: 23-Feb-2013]

[163] G. Carneiro, "Welcome to PyBindGen's documentation! — PyBindGen 0.15.0 documentation," 2009. [Online]. Available: http://pythonhosted.org/PyBindGen/. [Accessed: 23-Feb-2013]

[164] T. Nagy, *The Waf Book*. 2011, p. 73 [Online]. Available: http://docs.waf.googlecode.com/git/book_17/single.html. [Accessed: 22-Feb-2013]

[165] S. Holzner, *Ant: The Definitive Guide, 2nd Edition*, Second Edi. O'Reilly Media, 2005, p. 336.

[166] J. Calcote, *Autotools: A Practioner's Guide to GNU Autoconf, Automake, and Libtool*, 1st ed. No Starch Press, 2010, p. 360.

[167] T. Nagy, "waf - The meta build system - Google Project Hosting." [Online]. Available: http://code.google.com/p/waf/. [Accessed: 23-Feb-2013]

[168] B.W. Kernighan and D.M. Ritchie, *C Programming Language*, 2nd ed. Prentice Hall, 1988, p. 274 [Online]. Available: http://www.amazon.com/dp/0131103628

[169] I. Chivers and J. Sleightholme, *Introduction to Programming with Fortran: With*

Coverage of Fortran 90, 95, 2003, 2008 and 77, 2nd ed. 20. Springer, 2012, p. 643 [Online]. Available: http://www.amazon.com/dp/0857292323

[170] J. Hickey, A. Madhavapeddy, and Y. Minsky, *Real World OCaml*. O'Reilly Media, 2013, p. 450.

[171] A. Alexandrescu, *The D Programming Language*, 1st ed. Addison-Wesley Professional, 2010, p. 460 [Online]. Available: http://www.amazon.com/dp/0321635361

[172] A. Troelsen, *Pro C# 5.0 and the .NET 4.5 Framework*, 6th ed. Apress, 2012, p. 1560 [Online]. Available: http://www.amazon.com/dp/1430242337

[173] Nsnam, "Installation - Nsnam," 2012. [Online]. Available: http://www.nsnam.org/wiki/index.php/Installation. [Accessed: 22-Feb-2013]

[174] B. King, "GCC-XML." [Online]. Available: http://www.gccxml.org/HTML/Index.html. [Accessed: 23-Feb-2013]

[175] J. Fawcett, D. Ayers, and L.R.E. Quin, *Beginning XML, 5th Edition*, 5th ed. Wrox, 2012, p. 864 [Online]. Available: http://www.amazon.com/dp/1118162137

[176] W. von Hagen, *The Definitive Guide to GCC*, 2nd ed. Apress, 2006, p. 584 [Online]. Available: http://www.amazon.com/dp/1590595858

[177] R. Yakovenko, "pygccxml 0.6.9 : Python Package Index." [Online]. Available: https://pypi.python.org/pypi/pygccxml. [Accessed: 23-Feb-2013]

[178] C. Larman, *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*, 3rd ed. Prentice Hall, 2004, p. 736 [Online]. Available: http://www.amazon.com/dp/0131489062

[179] A.S. Tanenbaum, *Modern Operating Systems*, 3rd ed. Prentice Hall, 2007, p. 1104 [Online]. Available: http://www.amazon.com/dp/0136006639

[180] A. Evans, K. Martin, and M.A. Poatsy, *Technology In Action, Complete*, 10th ed. Prentice Hall, 2013, p. 656 [Online]. Available: http://www.amazon.com/dp/0133056228

[181] A. Robbins, *Unix in a Nutshell, Fourth Edition*, Fourth Edi. O'Reilly Media, 2005, p. 908 [Online]. Available: http://www.amazon.com/dp/0596100299

[182] J. Corbet, A. Rubini, and G. Kroah-Hartman, *Linux Device Drivers, 3rd Edition*, 3rd ed. O'Reilly Media, 2005, p. 640 [Online]. Available: http://www.amazon.com/dp/0596005903

[183] IEEE Standard for Information technology, "IEEE Std 802.3 - Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications - Section 1," vol. 2008, no. December. IEEE Computer Society, pp. 1–671, 2008 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5621025. [Accessed: 23-Feb-2013]

[184] IEEE Standard for Information technology, "IEEE Std 802.3 - Part 3 : Carrier Sense Multiple Access with Collision Detection (CSMA / CD) access method and Physical Layer specifications - Section 3," 2008.

[185] A.S. Tanenbaum, *Computer Networks (4th Edition)*. Prentice Hall, 2002, p. 912 [Online]. Available: http://www.amazon.com/Computer-Networks-Edition-Andrew-Tanenbaum/dp/0130661023. [Accessed: 22-Feb-2013]

[186] H. Zimmermann, "OSI Reference Model--The ISO Model of Architecture for Open Systems Interconnection," *IEEE Transactions on Communications*, vol. 28, no. 4, pp. 425–432, Apr. 1980 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1094702' escapeXml='false'/>. [Accessed: 04-Feb-2013]

[187] International Organization for Standardization, "ISO - International Organization for Standardization." [Online]. Available: http://www.iso.org/iso/home.html. [Accessed: 24-Feb-2013]

[188] R. Braden, "RFC 1122 - Requirements for Internet Hosts - Communication Layers." IETF, pp. 1–116, 1989 [Online]. Available: http://tools.ietf.org/html/rfc1122

[189] R. Braden, "RFC 1123 Requirements for Internet Hosts -- Application and Support." IETF, pp. 1–98, 1989 [Online]. Available: http://tools.ietf.org/html/rfc1123

[190] W. Stallings, *Data and Computer Communications*, 9th ed. Prentice Hall, 2010, p. 888

[Online]. Available: http://www.amazon.com/dp/0131392050

[191] D.E. Comer, *Computer Networks and Internets*, 5th ed. Addison-Wesley, 2008, p. 768 [Online]. Available: http://www.amazon.com/dp/0136061273

[192] K.R. Fall and W.R. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols (2nd Edition)*, 2nd ed. Addison-Wesley Professional, 2011, p. 1056 [Online]. Available: http://www.amazon.com/dp/0321336313

[193] J.F. Kurose and K.W. Ross, *Computer Networking: A Top-Down Approach*, 5th ed. Addison-Wesley, 2009, p. 864 [Online]. Available: http://www.amazon.com/dp/0136079679

[194] B. Forouzan, *Data Communications and Networking*, 5th ed. McGraw-Hill Science/Engineering/Math, 2012, p. 1264 [Online]. Available: http://www.amazon.com/dp/0073376221

[195] L.F.G. Soares, G.L. DE SOUZA FILHO, and S. Colcher, *Redes de computadores: das LANs, MANs, e WANs às redes ATM*. Campus, 1995, p. 705 [Online]. Available: http://books.google.com.br/books?id=4sayGQAACAAJ

[196] J. Postel, "RFC 792 - Internet Control Message Protocol." IETF, pp. 1–21, 1981 [Online]. Available: https://tools.ietf.org/html/rfc792. [Accessed: 05-Mar-2013]

[197] D.C. Plummer, "An Ethernet Address Resolution Protocol. ARPA RFC - 826." 1982 [Online]. Available: ftp://ftp.rfc-editor.org/in-notes/rfc826.txt

[198] J. Postel, "RFC 793 - Transmission Control Protocol," *RFC 793*, vol. 25, no. 793. IETF, pp. 1–85, 1981 [Online]. Available: http://www.ietf.org/rfc/rfc793.txt

[199] J. Postel, "RFC 768 - User Datagram Protocol." IETF, pp. 1–3, 28-Aug-1980 [Online]. Available: http://dl.acm.org/citation.cfm?id=RFC0768. [Accessed: 22-Feb-2013]

[200] Nsnam, "Packets — Model Library," 2012. [Online]. Available: http://www.nsnam.org/docs/release/3.15/models/html/packets.html. [Accessed: 24-Feb-2013]

[201] S.J. Leffler, W.N. Joy, R.S. Fabry, and M.J. Karels, "Networking Implementation Notes 4 . 4BSD Edition," Berkeley, 1993 [Online]. Available:

http://people.debian.org/~adamm/doc/smm/18.net/paper.pdf

[202] IEEE Standard for Information technology, “IEEE Std 1003.1 - Portable Operating System Interface (POSIX) Base Specifications, Issue 7.” pp. 1–3826, 01-Dec-2008 [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4694976’ escapeXml='false'/>. [Accessed: 24-Feb-2013]

APPENDIX A –CONCEPTS OF THE NS-3 SIMULATOR

This appendix gives an overview and describes the operations of the core components of the ns-3 software. In ns-3, the simulation core and models are implemented in C++ and built as a library which may be statically or dynamically linked to a C++ main program that defines the simulation topology and starts the simulator. It also exports nearly its entire API to Python using Python Wrappers [161] (generated using PyBindGen tool [162], [163]), allowing Python programs to import an *"ns-3 module"* in much the same way as a library is linked by an executable in C++. Figure 52 illustrates the ns-3 software organization introduced herein. The subsequent parts of this appendix describe the core parts of ns-3.

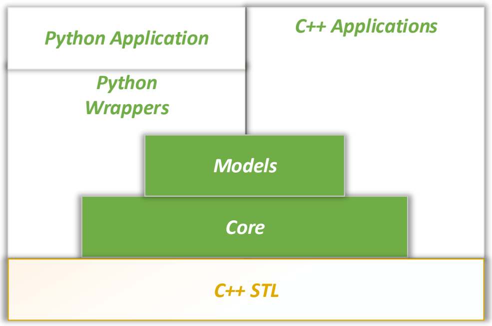


Figure 52: ns-3 Software Organization

A.1 WAF BUILD SYSTEM

As software is becoming increasingly complex, the process of creating software is becoming more complex [164]. Nowadays, software uses various languages, requires various compilers and the input data is spread into many files. Build systems make assumptions about the software it is trying to build, and are typically limited when it comes to processing other languages or different projects. For instance, Ant [165] is better suited than Make [166] for managing Java projects, but is more limited than Make for managing simple C projects [164]. The term *"build system"* is used to design the tools used to build applications.

The Waf framework [167] is somewhat different from traditional build systems in the sense that it does not provide support for a specific language. Rather, the focus is on supporting the major use cases encountered when working on a software project. As such, it is essentially a library of components designed to assist with the automatic compilation and installation of computer software. The default distribution contains various plugins for several programming languages (e.g. C [168], C++, Fortran [169], OCaml [170], D [171], C# [172], Java, Python, and so on) and different tools. The Waf build system is written in Python and is the main build system used on ns-3 projects.

The following figures illustrate the ns-3 build system. The Figure 53 demonstrates the steps involved during the build of the ns-3 project: *Download ns-3 code*, *Install required packages*, *Configure* and *Build*. To obtain the ns-3 code and install all the required packages, the ns-3 project indicates [173] and [96] respectively. The *Configure* step is necessary to analyze the system and verify that a minimum set of tools and libraries are presented, and to generate all the build scripts necessary for the last step. The *Build* step utilizes all scripts generated on the previous step to construct the ns-3 software, examples, tools and tests.

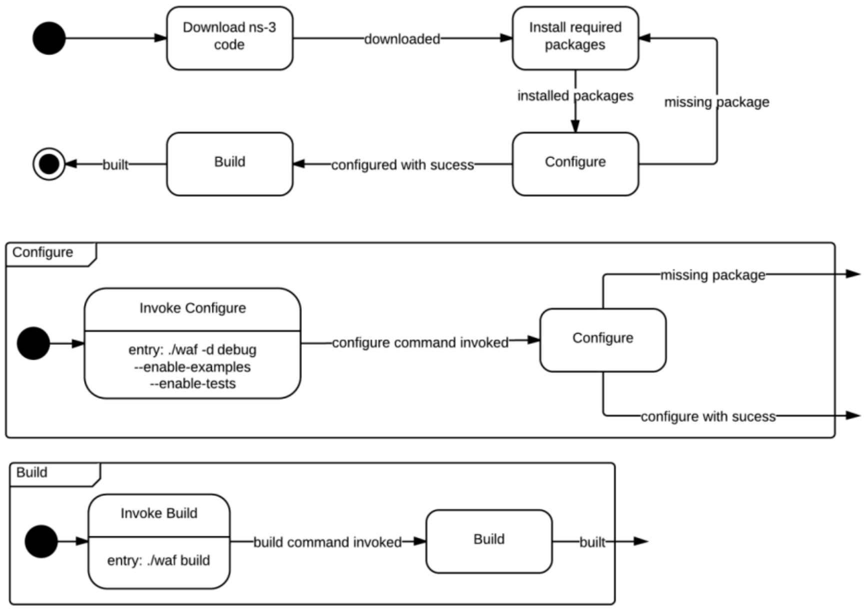


Figure 53: ns-3 Project Waf Built System

Figure 54 shows that the ns-3 project also utilizes the Waf build system to execute its examples or tests, as well as the programs representing scenarios or tests of a specific module, using the run command. The input for the run command is the name of the program built (Build step) and could also include the *"vis"* waf option that is utilized for PyViz support that is described in Chapter 4 (see Section 4.4).

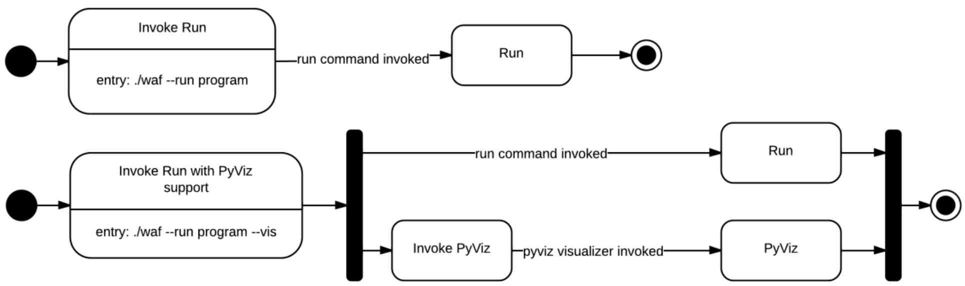


Figure 54: Running ns-3 Examples using the Waf Built System

Furthermore, the Waf build system can be used to generate Python wrappers for an ns-3 module using the PyBingGen tool, as previously mentioned. A Python wrapper enables seamless interoperability between C++ and Python programming languages. To compose ns-3 Python wrappers, PyBindGen utilizes the GCC-XML [174] extension that constructs an XML [175] description of the ns-3 C++ module using GCC [176] internal representation, as well as the pygccxml [177] that reads the GCC-XML generated file and provides a Python binding to easily navigate through C++ module declarations using Python classes. The generated bindings can be found in the *"bindings"* directory localized in the module directory. Figure 55 shows a compact state machine that illustrates the steps necessary to generate the module Python wrappers described here.

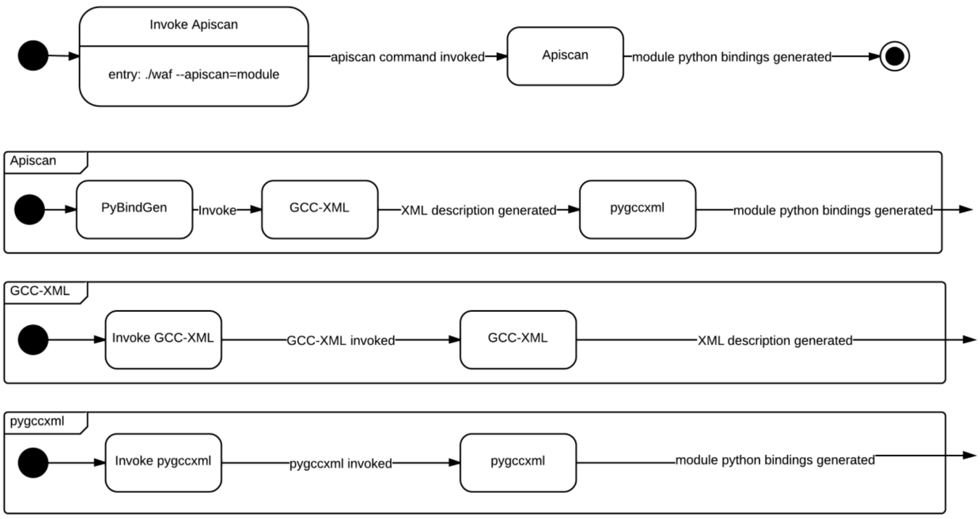


Figure 55: APIscan to Generate Module Python Wrappers

A.2 NODE

In ns-3, the basic computing device abstraction is called a *Node*. It provides methods for managing the representations of computing devices in simulations. Moreover, abstractions representing applications, protocol stacks and peripheral cards with their associated drivers could be added to perform useful tasks. The Figure 56 illustrates, using UML [178], the *Node* class and its possible internal components. This appendix describes some of these components next. The *EnergySource Container* and *EnergySource* components are part of the energy framework of ns-3 and are described in Chapter 4 (see Section 4.3).

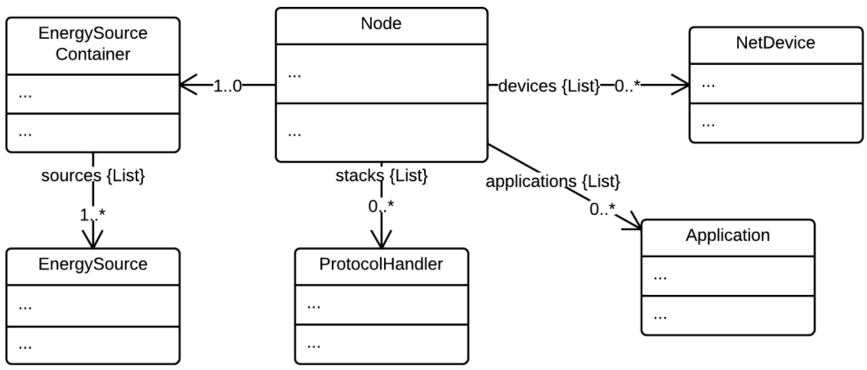


Figure 56: UML Class Diagram for Node Class

A.3 APPLICATION

Typically, computer software is divided into two broad classes: *System Software* and *User Applications*. System Software organizes various computer resources such as memory, processor cycles, disk, network, and so on, according to some computing model. A user would typically run a user application that acquires and uses the resources controlled by the system software to accomplish some goal.

Often, the line of separation between system and application software is made in the privilege level change that happens in operating system traps [179]. In ns-3, there is no real concept of operating system and especially no concept of privilege levels. However, it acts like an application just as software applications run on real computers to perform tasks.

In ns-3, the basic abstraction for a user program that generates some activity to be simulated is the *Application* class which runs on ns-3 *Nodes*. The Application class provides methods for managing application representatives of real version user-level applications in simulations. Developers are expected to create new applications using the *Application* class as the construction base. For instance, the *UdpEchoClientApplication* and *UdpEchoServerApplication* are ns-3 applications that simulate real client and server applications used to generate and echo simulated network packets. Figure 57 presents some examples of applications that simulate real applications and have been constructed using the *Application* class.

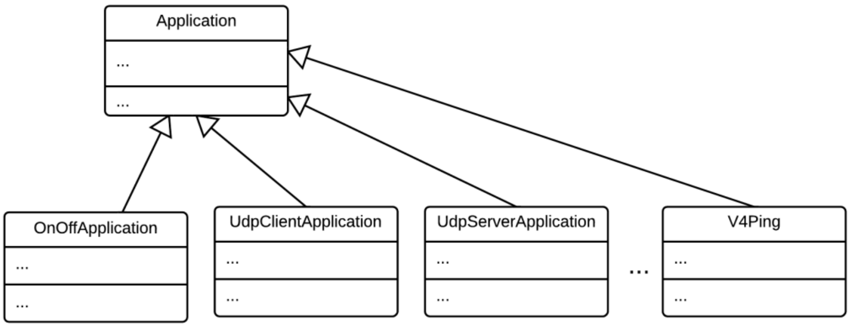


Figure 57: UML Class Diagram for Application Class

A.4 CHANNEL

In actuality, one can connect a computer to a network. Usually, the medium through which data flow in these networks are called 'channels'. For example, the Ethernet cable plugged into the plug on the wall to connect a computer to an Ethernet network is a communication channel. In the simulated world of ns-3, one connects a *Node* to an object representing a communication channel. Here the basic communication abstraction is called the channel and is represented by the class *Channel*.

A *Channel* may model something as simple as a wire. The specialized *Channel* can also model things as complicated as a large Ethernet switch, or a three-dimensional space full of obstructions in the case of wireless networks. Some examples of specialized channels are *WifiChannel*, *WimaxChannel*, and *SpectrumChannel*, which simulate the path over which information flows in WiFi, WiMAX and LTE networks, respectively. Figure 58 represents the channel class and some of its derivate classes.

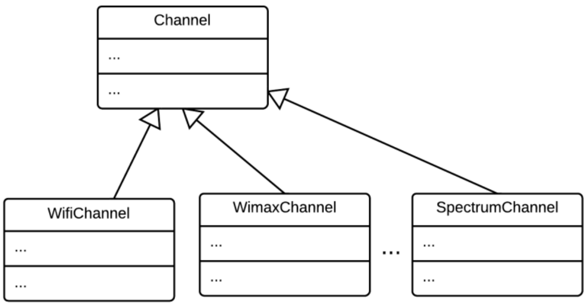


Figure 58: UML Class Diagram for Channel Class

A.5 NETDEVICE

In PC terminology, a hardware device that expands the capabilities of the host computer and does not form part of the core computer architecture is called peripheral card. A peripheral card is generally defined as any auxiliary device such as a computer mouse, keyboard or hard drive that connects to and works with the computer in some way. If the peripheral card implements some networking function, it is called a Network Interface Card or an NIC [180].

A NIC will not work without a software driver to control the hardware. In Unix [181] (or GNU / Linux), a piece of peripheral hardware is classified as a device. Devices are controlled using device drivers, and network devices (NICs) are controlled using network device drivers collectively known as net devices [182]. In Unix and GNU/Linux, these net devices are referred to by names such as "*eth0*".

In ns-3, the net device abstraction covers both the software driver and the simulated hardware. A net device is installed inside the Node in order to enable it to communicate with other *Nodes* during the simulation via *Channel* objects. The net device abstraction is represented by the class *NetDevice* that provides methods for managing connections among *Node* and *Channel* objects. Figure 59 exemplifies some net devices that represent real net devices and are specialized versions of the *NetDevice* class in the object-oriented programming sense. The *WifiNetDevice*, *WimaxNetDevice*, and *LteNetDevice* correspond to net devices used to connect hosts to WiFi, WiMAX and LTE networks respectively. They use, respectively, the *WifiChannel*, *WimaxChannel* and *SpectrumChannel* as channels. Other types of net devices exist, like *CsmaNetDevice,* which was designed to work with Ethernet or IEEE 802.3 networks. Each specialized net device can incorporate an object element representing the physical layer (e.g. *WifiPhy*, *WimaxPhy* and *LteSpectrumPhy*) that uses the channel to send and receive from its medium. The CSMA model based upon IEEE 802.3 standards [183], [184] does not precisely specify its physical components, such as wire types, signals or pin-outs, and thus the ns-3 does not include the equivalent of a 10BASE-T or 1000BASE-LX interface, for example.

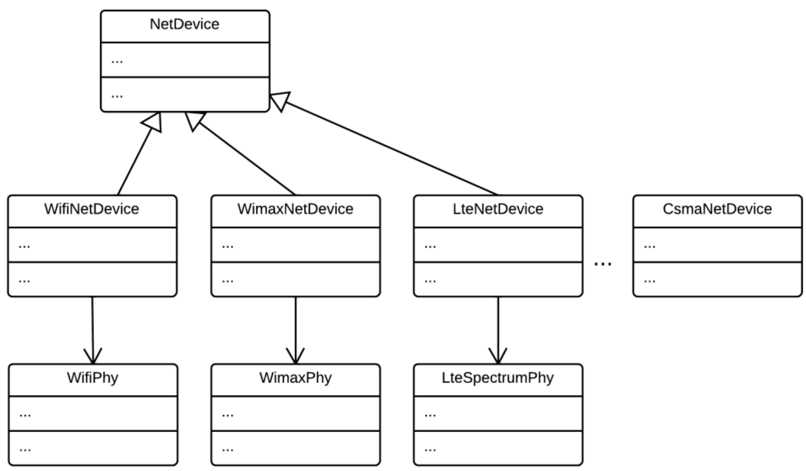


Figure 59: UML Class Diagram for NetDevice Class

A.6 PROTOCOLS AND PROTOCOLHANDLER

To reduce their design complexity, most networks are organized as a stack of layers or levels, each one built upon the one below it [185]. With layering, each layer is responsible for a different facet of communication. Layers are beneficial because they allow developers (often by different people with somewhat different areas of expertise) to address different portions of the system separately. The most frequently mentioned concept of protocol layering is based on a standard called the Open Systems Interconnection (OSI) model [186]. It suggests that seven logical layers may be desirable for the modularity of a protocol architecture implementation and it is defined by the International Organization for Standardization (ISO) [187]. Another layering model is the TCP/IP [188], [189] which uses fewer layers, varying from four to five [185], [188]–[195] (i.e. depending on reference). The ns-3 simulator adopts a TCP/IP 5-layer reference model [185] that is illustrated in Figure 60 with its layers and some examples of corresponding ns-3 elements for each layer.

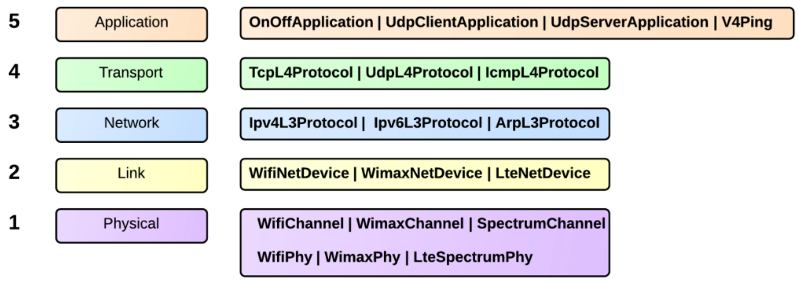


Figure 60: TCP/IP Layers and ns-3 Correspondent Objects

The ns-3 provides implementation support for some TCP/IP suite protocols that include IPv4, IPv6, ICMP [196], ARP [197], TCP [198], UDP [199] and so on. These protocols could be aggregated to the *Node,* registering a *ProtocolHandler* entry to them. The *ProtocolHandler* includes the protocol type and a handler function that must be called during reception packets. For instance, during a reception of layer-2 frames, these frames must be delivered to the right layer layer-3 protocol (e.g. IPv4) using a demultiplexer that dispatches it based on Ethernet type; in this case, the ns-3 *protocolType*, which is included in the layer-2 frame. The Figure 61 illustrates the steps required to deliver an incoming IP packet to the *Ipv4L3Protocol* that implements the IPv4 protocol. The next section explains the process involved during the transmission and reception of data packets in more detail.

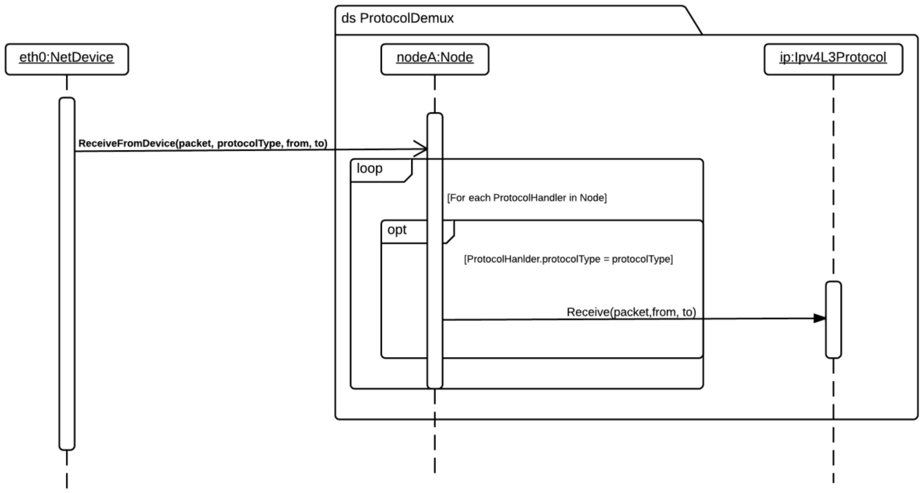


Figure 61: UML Sequence diagram representing the delivering of an incoming IP packet to layer-3 protocol

A.7 DATA FLOW

The data flow model represents the flow path involved in the transmission and reception of data using the ns-3 simulator. During the transmission and reception of application data, the ns-3 applications use a socket-like API that manages the access to the lower level transport protocol (e.g. TCP and UDP). The basic function of the transport layer is to accept data from the layer above it, split it up into smaller units if necessary, pass these on to the network layer, and ensure that the pieces all arrive correctly at the other end. The network layer controls the way in which packets are routed from source to destination. The next step involves the use of *NetDevice*, which represents the link layer, to transmit or receive frames using an appropriate channel. A data flow diagram is presented in Figure 62 and illustrates the model described. The socket API, *Packet* structure and the process of sending and receiving the packet are explained next. The process of packet fragmentation is not fully detailed along this this appendix and a further description can be read in [200]. The examples illustrated along this appendix use UDP as transport protocol, however the same logic can be used for other transport protocols (e.g. TCP) as well.

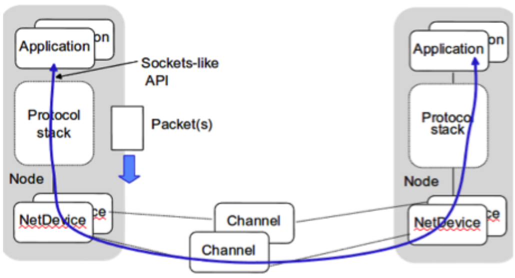


Figure 62: ns-3 Data Flow

A.7.1 PACKET

In a layered system, a unit of data specified for a protocol contained in a given layer and consisting of protocol control information and possibly user data is called PDU (Protocol Data Unit) [185] in general. Each layer has its own concept of a message object (a PDU) corresponding to the particular layer responsible for creating it. For example, if a layer 4 (transport) protocol produces a packet, it would properly be called a layer 4 PDU or transport PDU (Segment or TPDU) [192]. The PDU designation for each layer of TCP/IP 5-layer model is illustrated in the Figure 63.

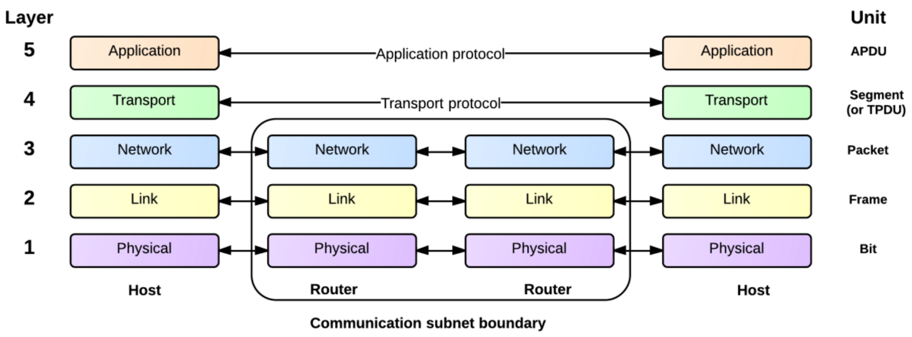


Figure 63: TCP/IP 5-layer Model and its PDUs.

In ns-3, the unit of data is the *Packet* (see Figure 64) that holds the data for each layer. The *Packet* object provides easy support for data fragmentation, defragmentation and concatenation. These objects contain a byte buffer, a *PacketMetadata*, a list of byte *Tags* and a list of packet *Tags*. The byte buffer stores the serialized content of the headers, and the payload and trailers added to the packet. The serialized representation of these headers is expected to match that of real network packets bit for bit which means that the content of the byte buffer is expected to be equal to the real version. The packet metadata describes the type of the headers and trailers which were serialized in the byte buffer. The tag lists are containers for extra items useful for simulation convenience, such as timestamps or flow identification. The *Packet* class deals with this requirement by storing a set of tags. The byte tags are used to tag a subset of the bytes in the packet byte buffer while packet tags are used to tag the packet itself. The main difference between these becomes clear when packets are copied, fragmented, and reassembled: byte tags follow bytes while packet tags follow packets. Moreover, byte tags cannot be removed and are expected to be written once and read many times, while packet tags are expected to be written once, read many times and removed exactly once.

The fundamental classes for adding to and removing from byte buffer are *Header* class and class *Trailer*. Every protocol header that needs to be inserted and removed from a *Packet* instance should derive from the abstract *Header* base class and implement the private pure virtual methods: *SerializeTo*, *DeserializeFrom*, *GetSerializedSize* and *PrintTo*. Basically, the first three functions are used to serialize and deserialize protocol control information to/from a buffer. The last function is used to define how the *Header* object prints itself onto an output stream. The HTR implementation provides a class derived from *Header* to reduce routing loops (see Section 3.10) that may occur during the transmission of data packets. The additional header is added during the routing of a packet to its destination and registers the nodes visited along the path.

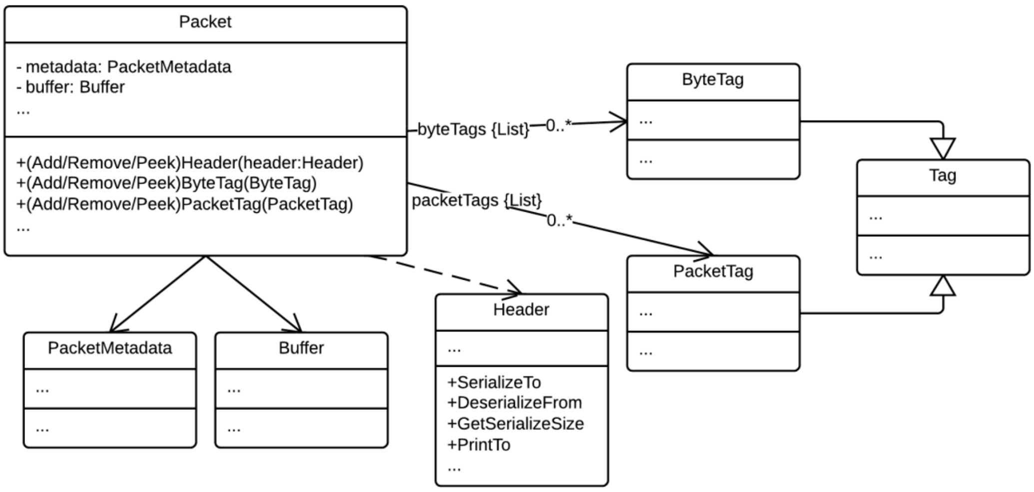


Figure 64: UML Class Diagram for Packet Classes

A.7.1 SOCKETS

Applications, whether p2p or client/server, need to express their desired network operations (e.g. make a connection, write or read data). This is usually supported by a host operating system using a networking application programming interface (API) [192]. The most popular network, API, is called sockets or Berkeley sockets, indicating where it was originally developed [201]. It was provided with UNIX distributions, software releases that pioneered the use of TCP/IP.

The ns-3 core provides a POSIX-like network API and strives to be close to the typical sockets API that application writers on Unix systems are accustomed to. POSIX [202] sockets are basically Berkeley sockets that have evolved over time, such that certain functions have deprecated or are even removed and replaced. The POSIX socket is much closer to a modern real system sockets API. The socket API abstraction is represented by the *Socket* class. The ns-3 native socket provides an interface for various types of transport protocols (e.g. TCP and UDP) via Socket derived classes (e.g. *TcpSocket* and *UdpSocketImpl*). Each socket object is generated using a correspondent socket factory that constructs it based on the protocol that it must provide. The UML diagram presented in Figure 65 shows that an application, for instance the *OnOffApplication*, creates the socket object using the correspondent factory and could use the socket object to bind the socket to a port (*Bind*), connect to a remote host (*Connect*), send (*Send*) or receive (*Receive*) packets to/from it or close the connection (*Close*). The process of sending and receiving data flow is explained next.

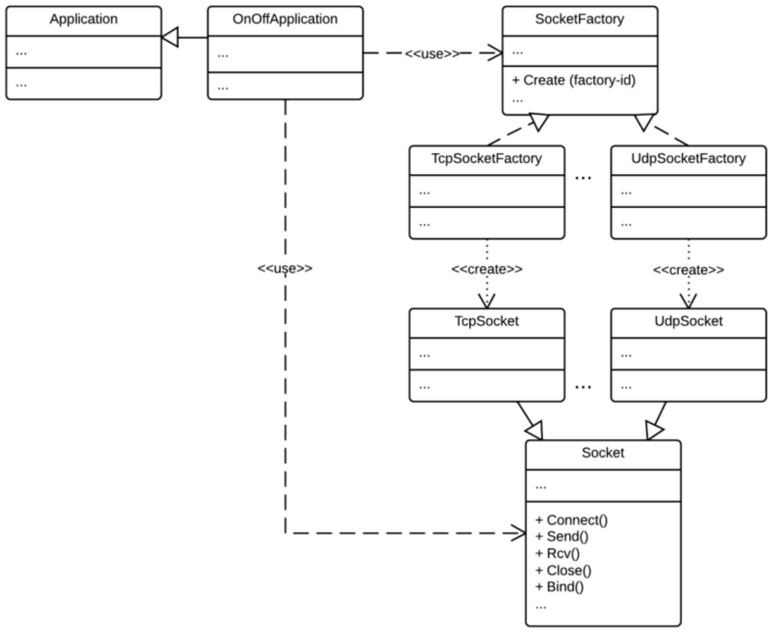


Figure 65: UML Class Diagram for Socket Creation and Use by an Application

A.7.2 SENDING A PACKET

For transmitting packets with valued data, ns-3 utilizes the following steps that are specified in Figures Figure 66, Figure 67, Figure 68 and Figure 69:

1. The application has previously created a socket (e.g. *UdpSocket*) and calls the *Send* method (see Figure 66) passing the packet with data, destination to reach and socket control flags (if necessary).
2. The *Send* method sets the proper source and destination address of the packet, handles socket calls such as *Bind* or *Connect*, queries the IPv4 routing system to find an available route to the destination address (*RouteOutput*) and calls the *Send* method of *UdpL4Protocol,* passing on the packet, source address, destination address, source port, destination port and route obtained. If the destination address is a (sub)broadcast or multicast address the query for the route is not necessary. The packet is dropped if the destination address is not one of those options or if no route is encountered to reach the destination. This step is modeled in the *"Udp Socket Switch"* UML sequence diagram (see Figure 67). More details

about the *RouteOutput* process are detailed in Chapter 4 (see Section 4.2).

3. *UdpL4Protocol* is where the socket independent protocol logic for UDP is implemented. The same logic could be used for the *TcpL4Protocol*, which implements the logic of the TCP protocol. For the example illustrated below, the UDP protocol is used. The *Send* method of *UdpL4Protocol* adds the UDP header, initializes checksum and sends the packet to the IPv4 layer represented by *Ipv4L3Protocol*.
4. The *Ipv4L3Protocol* adds the IP header and sends the packet to the appropriate *Ipv4Interface* based on the route that was passed down from the UDP layer. The purpose of the *Ipv4Interface* is to provide address-family specific information about an interface (i.e. *NetDevice*). The route, obtained in step 2, contains the address of the output interface, thus the *Ipv4L3Protocol* must send the packet to the *Ipv4Interface* that contains the same output address. *Ipv4L3Protocol* uses the route obtained if the destination address is not a (sub)broadcast or multicast address and if the route contains a valid gateway address to transmit. The packet is dropped if the route is invalid or if the destination address is not one of those options. This step is illustrated in *"IPv4 Send Switch"* (see Figure 68).
5. The *Ipv4Interface* checks if the destination address is intended for a loopback interface or a local interface, or if it needs to be transmitted elsewhere. For the first case, the packet may be broadcasted to the *NetDevice*. The second alternative delivers the packet to the *Ipv4L3Protocol* for reception. In the last case, the *Ipv4Interface* looks up the MAC address [190] if the ARP protocol is supported on *NetDevice*. If supported, it first searches for a hit at ARP cache, initializes an ARP request and then waits for a reply if there is no hit on its cache. The *Ipv4Interface* sends the packet to the *NetDevice,* passing on the MAC address as argument or as a broadcast address on the assumption that ARP is not supported. This step is depicted in "Ipv4 Interface Send Switch" (see Figure 69).
6. The *NetDevice* transmits the packet using the MAC address provided.

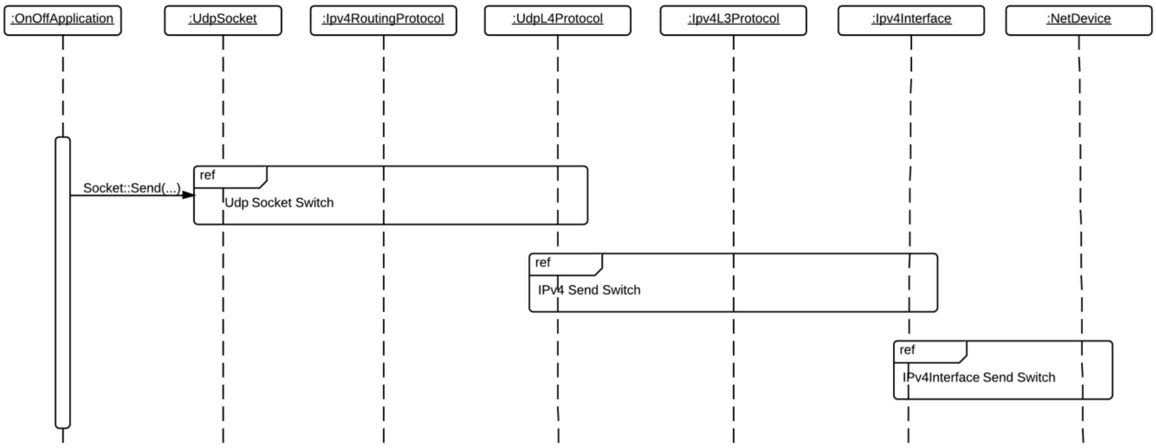


Figure 66: Sending a Packet

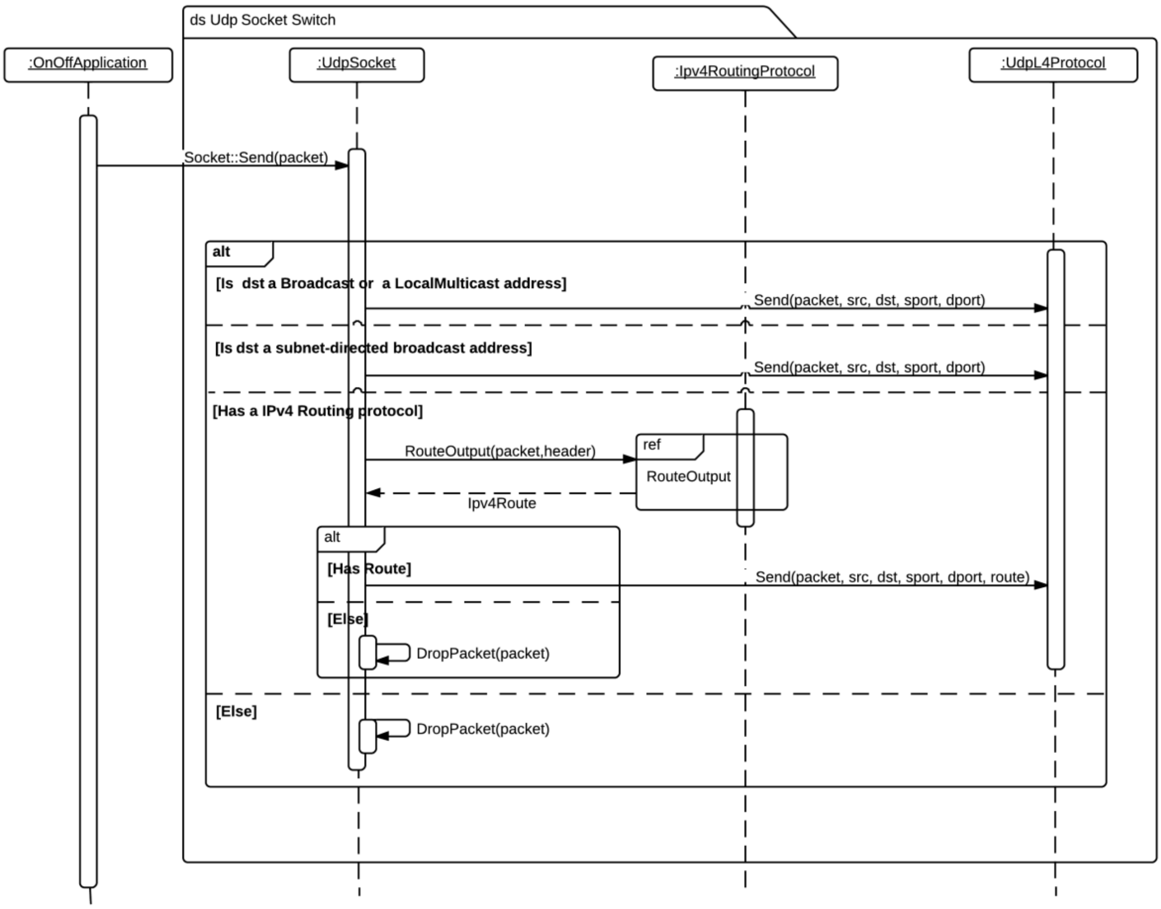


Figure 67: Udp Socket Switch

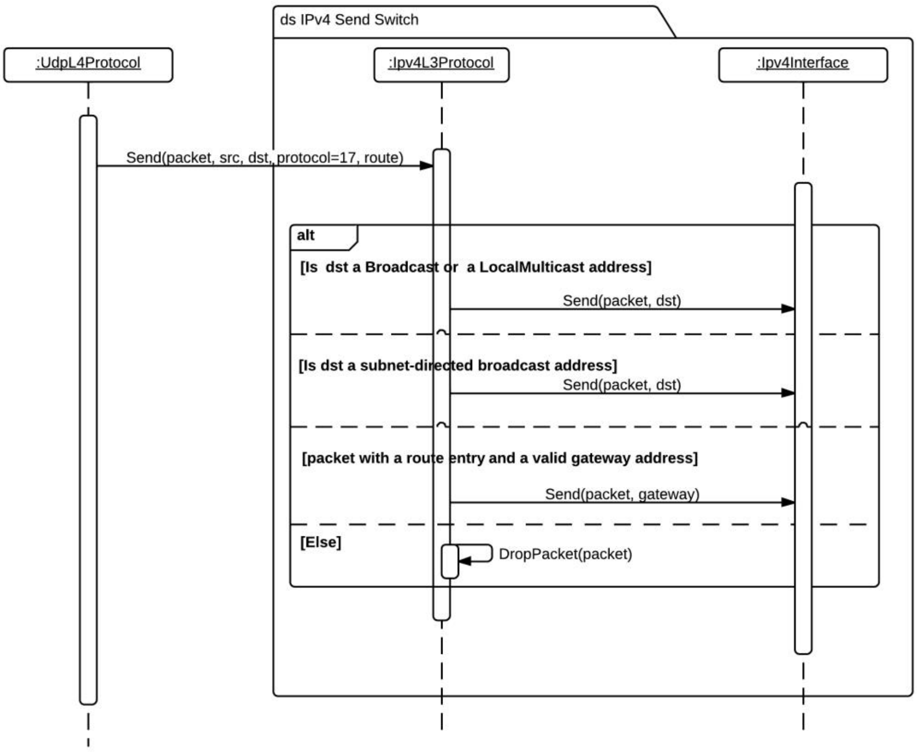


Figure 68: IPv4 Send Switch

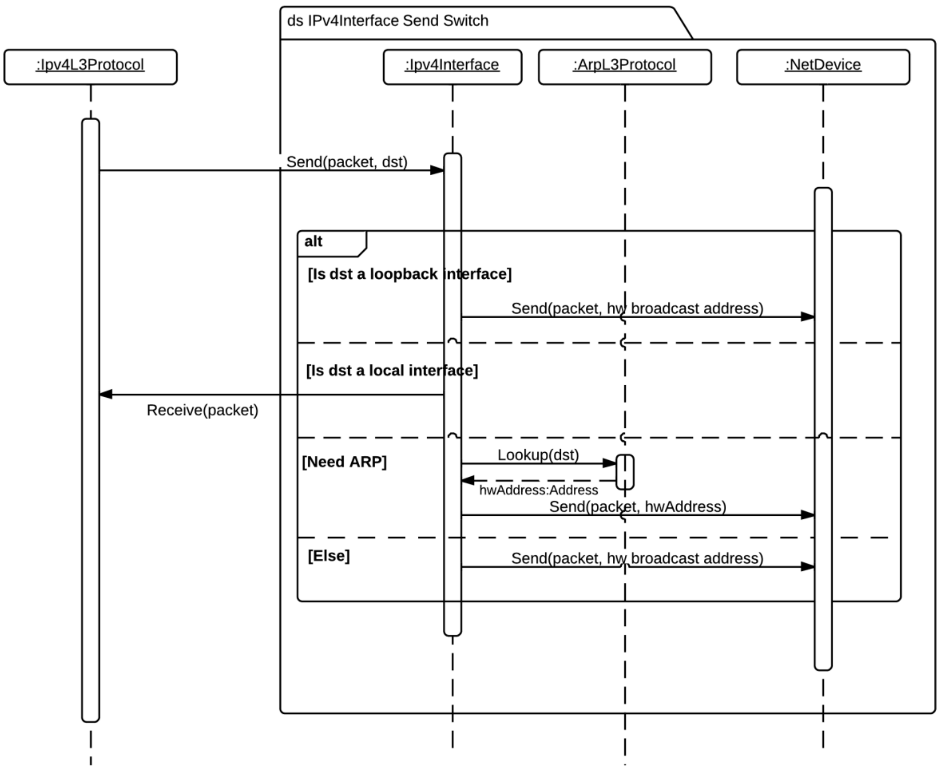


Figure 69: IPv4 Interface Send Switch

A.7.3 RECEIVING A PACKET

During packet reception, the *NetDevice* delivers the packet to the *"ProtocolDemux"* procedure that is processed on *Node*. This one looks up, based on protocol number and device, whether there is a protocol handler to be called. For the illustrated case (see Figure 70), the packet is delivered to the *Ipv4L3Protocol* since its protocol number corresponds to an IPv4 packet. The *IPv4L3Protocol* removes the IP header, checks the checksum (if implemented) and passes the packet using the *"RouteInput"* method of the Ipv4 routing system, which is implemented by a class derived from Ipv4RoutingProtocol class (e.g. HTRRoutingProtocol).

The *RouteInput* method decides whether the incoming packet may be forwarded elsewhere or locally delivered. The forwarding of packets utilizes a unicast or a multicast approach, which is performed using the *IpForward* and *IpMulticastForward* methods respectively, which are contained in the *Ipv4L3Protocol*. The *Ipv4L3Protocol* forwards the packet using the *"IPv4 Interface Send Switch"* mechanism based on the route obtained during the *RouteInput* method. If the packet is intended to be locally delivered, the *LocalDeliver* method of *Ipv4L3Protocol* is requested. The *LocalDeliver* method uses the *GetProtocol* method to identify the layer-4 protocol in use, such as the *UdpL4Protocol*, that it must deliver to the packet. The *Receive* method of the chosen protocol is called, removes the UDP header and tries to find a corresponding attached UDP socket indexed by an application destination port.

The *UdpL4Protocol* uses the method *ForwardUp* to deliver the packet to the corresponding socket, such as the *UdpSocket,* which waits for an application call to the method *RecvFrom,* which is the one to transfer the packet to the target application. The Figure 70 illustrates the reception of a packet including the steps described here. For further explanation regarding the IP forwarding and local delivery systems consult Section 5.4 of [185].

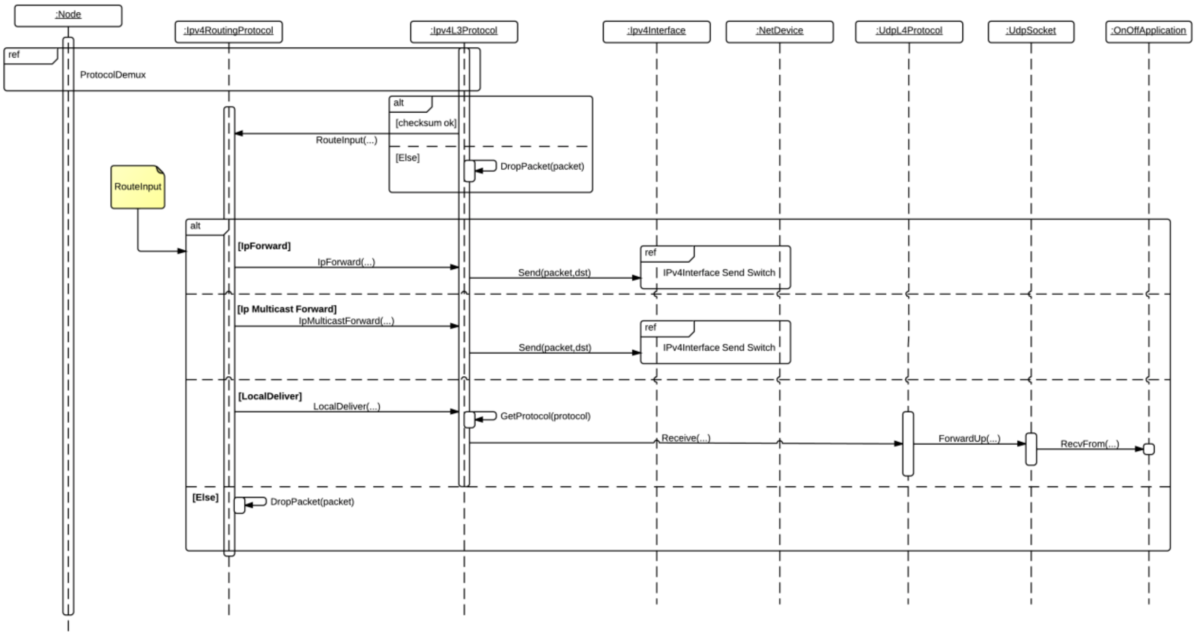


Figure 70: Receiving a Packet