NCTUns Simulation Tool for WiMAX Modeling

(Invited Paper)

Shiang-Ming Huang
Department of
Computer Science
National Chiao Tung University
Taiwan
smhuang@cs.nctu.edu.tw

Ya-Chin Sung
Department of
Computer Science
National Chiao Tung University
Taiwan
ycsung@csie.nctu.edu.tw

Shie-Yuan Wang
Department of
Computer Science
National Chiao Tung University
Taiwan
shieyuan@csie.nctu.edu.tw

Yi-Bing Lin
Department of
Computer Science
National Chiao Tung University
Taiwan
liny@liny.csie.nctu.edu.tw

ABSTRACT

This paper describes the NCTUns simulation tool that facilitates the WiMAX studies of network behaviors and performance analysis. We compare NCTUns with ns-2, QualNet, and OPNET Modeler. The design paradigms of these tools are elaborated. Then we enumerate the WiMAX-related functions that can be simulated by these tools. The performance of NCTUns is studied by an Ethernet network simulation and a WiMAX network simulation.

Categories and Subject Descriptors

D.2.8 [**Software Engineering**]: Metrics—*performance measures*

General Terms

Performance

Keywords

NCTUns, simulation, WiMAX

1. INTRODUCTION

NCTUns is a simulation tool developed by National Chiao Tung University [1]. NCTUns simulates the hardware characteristics of network devices (e.g., hubs or switches), the protocol stacks employed in these devices (e.g., the bridge-learning protocol used in a switch), and the execution of application programs on these devices. Moreover, this tool provides network utility programs for configuring network topologies, specifying network parameters, monitoring traffic flows, gathering statistics about a simulated network, etc.

The current released version NCTUns 4.0 implements the *Worldwide Interoperability for Microwave Access* (WiMAX) [9] protocol stacks to facilitate the functional and performance studies of the WiMAX system. WiMAX is a broadband wireless system which offers packet switched services for fixed, nomadic, portable and mobile accesses. WiMAX utilizes many advanced technologies in the *physical* (PHY) and the *medium access control* (MAC) layers to provide high spectrum efficiency.

Recently, several WiMAX simulators have been developed to provide a preliminary analysis for advanced WiMAX system development, including the open source tool ns-2 [2], and commercial softwares such as QualNet [8] and OPNET Modeler [4]. The WiMAX module for ns-2 can be downloaded at [10]. This paper compares these simulation tools. Section 2 briefly introduces the WiMAX technology. Section 3 describes the designs of NCTUns, ns-2, QualNet, and OPNET and compare the WiMAX-related functions implemented in these tools. Section 4 conducts performance measurements for both NCTUns and OPNET, and Section 5 concludes this paper.

2. OVERVIEW OF WIMAX

The IEEE 802.16 standard [11] or WiMAX defines the PHY and MAC layers to support multiple services with point-to-multipoint and mesh broadband wireless access. The point-to-multipoint mode defines one-hop communication between a base station (BS) and a subscriber station (SS), while the mesh mode allows traffic to be directly exchanged and forwarded among neighboring SSs. IEEE 802.16 is initially designed as an access technology for wireless metropolitan area network (WMAN). The first specification IEEE 802.16-2004 targets on fixed and nomadic accesses. In IEEE 802.16e-2005 amendment, the IEEE 802.16e system (also called Mobile WiMAX) further provides functions to facilitate mobile accesses. We introudce the functions of MAC and PHY layers in the following subsections. Details of WiMAX technology can be found in [12].

WICON 2007, October 22-24, Austin, USA
Copyright © 2007 978-963-9799-12-7
DOI 10.4108/wicon.2007.2270

2.1 The Media Access Control Layer

There are three sublayers in IEEE 802.16 *Media Access Control* (MAC) layer: service-specific *convergence sublayer* (CS), the MAC common part sublayer, and the security sublayer.

The service-specific CS performs packet classification, header suppression, and converts packets between the upper layer and the MAC layer. The IEEE 802.16 currently supports packet CS and ATM CS to interface with IP and ATM protocol layers, respectively. In IEEE 802.16, the connections between the SSs and the BSs can be identified with unique *connection identifications* (CIDs). The packet CS may check the IP or TCP/UDP header of a packet to determine its CID. Besides the CID mapping, the CS may perform the optional payload header suppression to eliminate the redundant parts of the packets during the transmission over the air interface.

The MAC common part sublayer provides the medium access, connection management, and QoS functions that are independent of specific CSs. After the packets are processed by the CS, the MAC common part may perform *automatic repeat request* (ARQ) for retransmitting lost packets. ARQ is optional in IEEE 802.16 but is mandatory for IEEE 802.16e.

In IEEE 802.16, QoS functions are implemented in the MAC common part sublayer. Several service classes are defined to satisfy various QoS requirements. For example, a VoIP connection is often associated with "*unsolicited grant service*" (UGS) to support constant bit-rate (CBR) or CBR-like flows with constant bandwidth allocation. According to the QoS associated, the BS schedules radio resources with various schediling disciplines, such as Round Robin and *First-in, First-out* (FIFO).

The security sublayer provides privacy and protections through encryption, decryption, and authentication. In IEEE 802.16, an SS is requested to perform the authentication and authorization before attaching to a WiMAX network. During the authorization procedure, the SS negotiates with the BS to generate the session key. To perform packet encryption and decryption, each connection is linked with a *security association* (SA), which contains the security information and settings such as encryption keys. Packet encryption and decryption are exercised based on the information in the SA.

Before accessing the WiMAX network, an SS should perform a complete spectrum search and synchronize the time and frequency with a BS through the ranging procedure. Then the SS starts the network entry procedure to negotiate the capabilities with the BS and performs authorization process to generate the keys used between the SS and the BS. Finally, the SS obtains an IP address from the BS, and establishes data connections with the BS.

2.2 The Physical Layer

IEEE 802.16 defines several *Physical* (PHY) layer specifications for different frequency ranges and applications. For example, *Orthogonal frequency division modulation* (OFDM) is used for *non-line-of-sight* (NLOS) operations in the frequency bands below 11 GHz. By extending the OFDM technology, *Orthogonal frequency division multiple access* (OFDMA) allows one channel to be shared by multiple users.

The IEEE 802.16 standard defines a set of adaptive modulation and coding rate configurations that can be used to trade off data rate against system robustness under various wireless propagation and interference conditions. The allowed modulation types are *binary phase shift keying* (BPSK), *quadrature phase shift keying* (QPSK), *16-quadrature amplitude modulation* (16QAM), and 64QAM [11].

Several duplexing technologies are provided in IEEE 802.16. In time divison duplex (TDD), a WiMAX frame consists of a downlink (DL) subframe and an uplink (UL) subframe and a short transition gap is placed between the DL and UL subframes for receive and transmission transitions in the radio. The gap between the downlink burst and the subsequent uplink burst is called *transmit/receive transition gap* (TTG). The gap between the uplink and the subsequent downlink is called *receive/transmit transition gap* (RTG).

The duration of OFDM symbol includes the useful symbol time and a prefix. In OFDM, all users within the same cell or sector use orthogonal subcarriers to carry the OFDM symbols. The OFDM symbol uses a fixed-length *cyclic prefix* (CP) to counteract the intersymbol interference. The ratio of the CP length to the useful symbol time is defined as the guard interval, which is used by the receiver to collect signals from multiple paths and improve system performance.

3. SIMULATORS FOR WIMAX

This section compares NCTUns with ns-2, QualNet, and OPNET Modeler. We compare the design paradigms of these simulators in section 3.1. Section 3.2 elaborates on the WiMAX-related functions implemented in these simulation tools.

3.1 Design Paradigm

NCTUns, ns-2, QualNet, and OPNET Modeler are object oriented simulation tools. They are implemented in a scalable paradigm, which allows users to add modules for their proprietary protocols. NCTUns provides *application programming interfaces* (APIs) for users to register the developed modules to the simulator. Ns-2 requires users to edit TCL scripts to add new modules implemented in C++. QualNet provides Model Libraries to construct user-defined network objects. OPNET Modeler allows users to add new modules by building library modules. NCTUns is executed on Fedora 7 with Linux 2.6.21 kernel. Ns-2 is run on UNIX Systems. QualNet, and OPNET Modeler supports both Microsoft Windows and UNIX systems.

The simulation engines of these tools are implemented in C-based languages. NCTUns is implemented in the C++ language. Ns-2 consists of a C++ engine combined with *Object-oriented TCL* (OTCL) [3]. Objects, such as protocols, links, nodes, and pakets, are implemented in C++, and are connected and controlled by using OTCL. QualNet is designed with parallel discrete-event simulation capability provided by Parsec, which is a C-based simulation language for sequential and parallel execution of discrete-event simulation models [5]. The engine of OPNET Modeler is designed with finite state machine model using Proto-C [7], a modified C language.

Ns-2, QualNet, and OPNET Modeler do not support the standard UNIX POSIX API system calls [6]. As such, existing and to-be-developed real-life application programs cannot be directly run with the simulation engine to generate traffic for a simulated network. Instead, real applications must be modified or re-implemented to use the internal API provided by the simulator (if there is any) and be compiled with the simulator to form a single, large, and complex program. Since the applications must be re-implemented in the simulation programs, another validation is required to ensure that the simulated applications are consistent with the real applications.

The above issue is resolved by NCTUns. NCTUns utilizes a kernel re-entering simulation methodology [13], which employs tunnel network interfaces available on most UNIX machines to simulate the TCP/IP network. From the kernel's point of view, the functions of a tunnel network interface are no different from those of an ordinary network interface. A real-life application program can exchange packets with the destination host through tunnel network interfaces, just as if these packets were sent to or received from normal network interfaces. Using this methodology, real-life applications can be run over the real protocol stacks without any modification of the application program.

The network simulators may utilize traffic generators to generate packet events. NCTUns supports both real-life application traffic (described above) and self-developed traffic generators. Neither NCTUns nor ns-2 provide *graphical user interface* (GUI) to configure the traffic generator and require the users to write scripts to specify the parameters such as packet length, packet inter-arrival time, and the distribution of the traffic. On the other hand, both QualNet and OPNET provide GUI for users to configure the parameters of the traffic generators.

The network simulators often provide GUI to present visualized simulation results. The visualized tool provided by NCTUns is coded in C++ with Qt library. The visualized tool provided by ns-2 is TCL-based. The visualized tool provided by QualNet is coded in JAVA. The visualized tool provided by OPNET is implemented in C++.

3.2 WiMAX Function Implementations

Table 1 lists the WiMAX functions implemented in NCTUns, ns-2, QualNet, and OPNET Modeler. All of these tools support fixed WiMAX simulation under point-to-multipoint topology. NCTUns supports mesh topology simulation. Mobile WiMAX simulation is only supported by QualNet when this paper is written.

In the MAC layer, basic MAC functionalities such as scheduling, QoS control, network entry and initialization procedure, etc., are implemented in these four simulators. Classification function and payload header suppression are optional in Packet CS. The classification function is supported by ns-2, QualNet, and NCTUns. The payload header suppression function is implemented in ns-2, QualNet, and OPNET Modeler. ARQ is optional in data communication, and is supported by OPNET. In the PHY layer, OFDM technology simulation is supported in NCTUns, QualNet, and OPNET Modeler. OFDMA technology simulation is provided by ns-2, QualNet, and OPNET Modeler. The modulation scheme can be easily supported by adding modules, and the modulation supports listed in table 1 are the schemes currently supported in the simulation tools.

4. PERFORMANCE EVALUATION

This section reports the simulation performance of NCTUns. Section 4.1 demonstrates an Ethernet network simulation. Section 4.2 specifies a WiMAX network simulation.

4.1 Ethernet Network Simulation

Table 1: WiMAX Functions for (A) NCTUns, (B) ns-2, (C) QualNet, and (D) OPNET Modeler

				A	B	C	D
802.16-2004 (fixed WiMAX)				O	O	O	O
802.16e-2005 (mobile WiMAX)				X	X	O	X
point-to-mutlipoint topology				O	O	O	O
mesh topology				O	X	X	X
MAC	data plane	Packet CS	Classification	O	O	O	X
			payload header suppression	X	O	O	O
		ARQ mechanism		X	X	X	O
		scheduling services		O	O	O	O
	control plane	bandwidth allocation and request mechanisms		O	O	O	O
		contention resolution		O	O	O	O
		network entry and initialization		O	O	O	O
		ranging		O	O	O	O
		QoS		O	O	O	O
PHY	type	OFDM		O	X	O	O
		OFDMA		X	O	O	O
	modulation scheme	BPSK		O	X	X	X
		QPSK 1/2		O	O	O	O
		QPSK 3/4		O	O	O	O
		16QAM 1/2		O	O	O	O
		16QAM 3/4		O	O	O	O
		64QAM 1/2		X	X	X	O
		64QAM 2/3		O	O	O	O
		64QAM 3/4		O	O	O	O

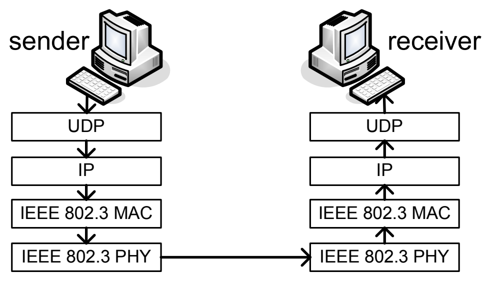
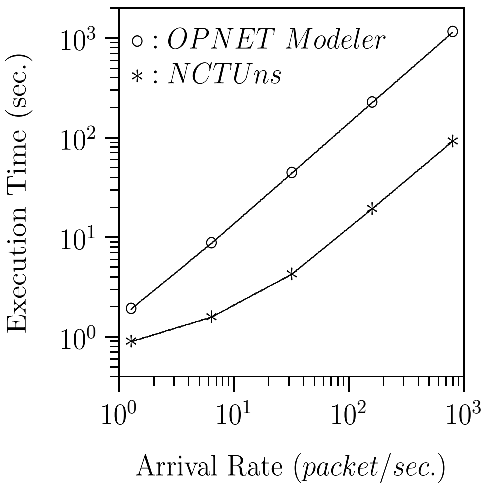
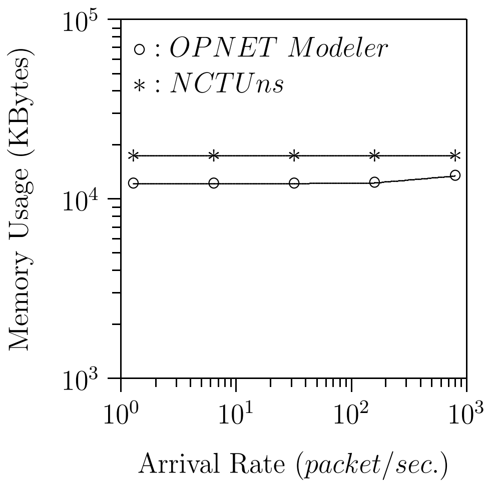


Figure 1: The Architecture of the Simulated Ethernet Network Protocols



(A) Execution Time Comparisons



(B) Memory Usage Comparisons

Figure 2: Performance Comparisons of NCTUns 4.0 and OPNET Modeler 12.0 in CPU Execution time of the Ethernet Network Simulation

Figure 1 illustrates the architecture of the simulated Ethernet network. The simulated system consists of a single UDP sender and a single UDP receiver. The distance between the UDP sender and the UDP receiver is 500 meters and the bandwidth of the Ethernet link between the UDP sender and the UDP receiver is 10 Mbps. The size of the transmitted packet is 1400 bytes, and the simulation period is 3000 seconds.

Figure 2 (A) shows the CPU execution time of the simulated case on OPNET Modeler 12.0 and NCTUns 4.0. The figure shows that the CPU execution time of OPNET Modeler is 11.64 times longer than that of NCTUns when the packet transmission rate is 160 packets per second. In OPNET Modeler, the CPU execution time increases from 1.9 seconds to 44 seconds (i.e., increases by 23.18 times) when the packet transmission rate increases from 1.28 packets per second to 32 packets per second (i.e., increases by 25 times). In NCTUns, the CPU execution time increases from 0.9 seconds to 4.26 seconds (i.e., increases by 4.73 times) under the same parameter setups as those of OPNET Modeler.

Figure 2 (B) shows the performance of the memory usage during the simulation. It is shown that the memory usage of NCTUns is 1.43 times that of OPNET Modeler when the packet transmission rate is below 32 packets per second in the simulated Ethernet network case. The memory usage of the simulated case on both NCTUns and OPNET Modeler increases insignificantly as the UDP packet transmission rate increases. Specifically, the memory usage in OPNET Modeler increases from 12,279 KB to 13,423 KB (i.e., increases by 1.09 times) when the packet transmission rate increases from 160 packets per second to 800 packets per second (i.e., increases by 5 times). In NCTUns, the memory usage increases from 17,396 KB to 17,415 KB (i.e., increases by 1.001 times) under the same simulated case as OPNET Modeler.

The performance of NCTUns and ns-2 were evaluated in [13]. Both tools support playback function by recording the trace file of the packet transmission during the simulation. It was reported that the memory usage of the two simulation tools are very close to each other in the Ethernet network simulation, regardless whether the trace file is generated or not. The execution time performance of NCTUns is slightly better than that of ns-2 when these tools are requested to generate the trace file. However, ns-2 outperforms NCTUns in terms of execution time when the trace file option is disabled.

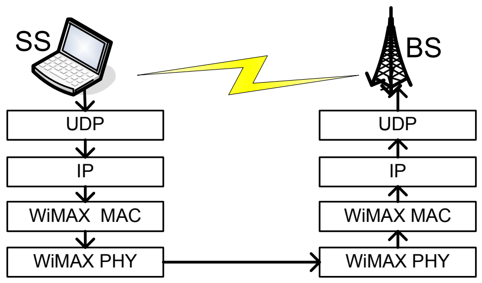


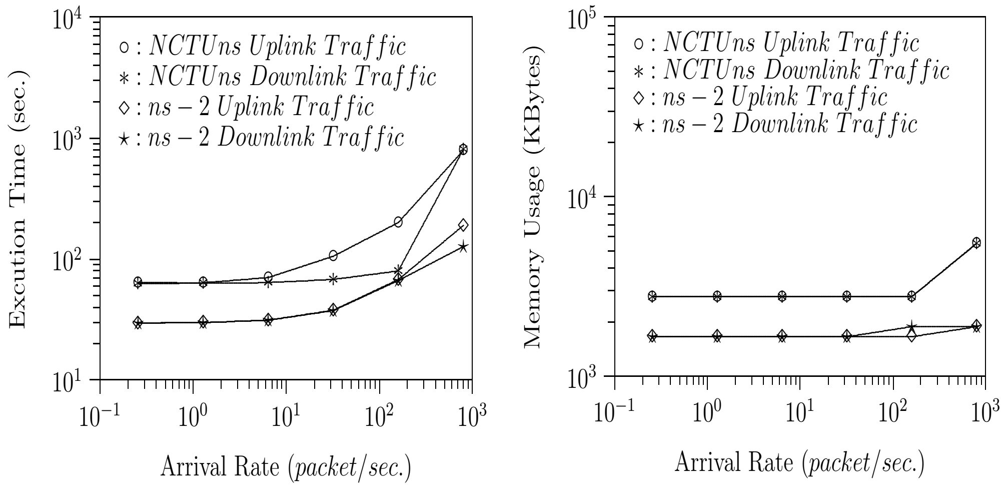
Figure 3: The Architecture of the Simulated WiMAX Network Protocols

4.2 WiMAX Network Simulation

This section evaluates the performance of a WiMAX network simulation. Figure 3 illustrates the architecture of the simulated WiMAX network, and Table 2 lists the configurations used. The simulated system consists of a single SS and a single BS. The SS sends 1400-byte UDP packets to the BS. The QoS applied in the MAC common part sublayer is UGS, and the scheduler employs Round Robin algorithm in radio resource allocation. The PHY technology utilizes OFDM with TDD and the modulation scheme is 64QAM 3/4. The simulation period is 1000 seconds.

Figure 4 shows the CPU/memory performance of NCTUns and ns-2 in WiMAX network simulation. We cannot access WiMAX modules of OPNET 12.0. To accurately compare OPNET Modeler with NCTUns, the reader can measure OPNET WiMAX simulation using the parameters described in Table 2 and investigate its CPU/memory performance with respect to the NCTUns measures in Figure 4.

Figure 4 (A) shows the evaluated CPU execution time of



(A) Execution Time Comparisons (B) Memory Usage Comparisons

Figure 4: Performance of the Uplink/Downlink Traffic Simulations in NCTUns and ns-2

Table 2: Parameter Setups for the WiMAX Simulation

Parameter			Value
system	BS antenna height		80m
	SS antenna height		15m
	BS-SS distance		100m
input traffic	QoS class		UGS
	traffic		UDP
	input direction		UL/DL
MAC	data plane	UL grant scheduler	Weighted Round Robin
		grant weight	same for all SS
		DL scheduler	Round Robin
		ARQ	disabled
	control plane	ranging/scan interval	infinity
PHY	type		OFDM
	modulation scheme		64QAM 3/4
	the normal channel bandwidth		20MHz
	number of used subcarriers		192
	sampling factor		8/7
	guard interval		1/4
	frame duration		10ms
	downlink to uplink ratio		1
	number of sectors		1
	TTG		$5\mu s$
	RTG		$5\mu s$
medium	channel error		None

the WiMAX simulations on NCTUns and ns-2. It is shown that the CPU execution time of NCTUns is longer than that of ns-2. Specifically, the CPU execution time of NCTUns is 2.16 times that of ns-2 for both the uplink and downlink traffic simulations when the packet transmission rate is below 1.28 packets per second. Moreover, the CPU execution time of NCTUns is 4.18 times that of ns-2 for the uplink traffic simulation and 6.36 times for the downlink traffic simulation when the packet transmission rate is 800 packets per second.

Figure 4 (B) shows the performance of NCTUns and ns-2 with respect to the memory usage during the simulations. It is shown that the memory usage of NCTUns is larger than that of ns-2. Specifically, the memory usage of NCTUns is 1.67 times that of ns-2 for both the uplink and downlink traffic simulations when the packet transmission rate is below 32 packets per second. Moreover, the memory usage of NCTUns is 2.91 times that of ns-2 for both the uplink and downlink traffic simulations when the packet transmission rate is 800 packets per second.

We observe that the CPU and memory performance of NCTUns for the WiMAX simulations are a little worse than that of ns-2. The reason is that NCTUns supports real-life application program simulations, which causes overhead on CPU execution time and memory usage.

5. CONCLUSIONS

This paper investigates the simulation tool NCTUns for Wi-MAX modeling. It also compares NCTUns with ns-2, QualNet, and OPNET Modeler.

The design paradigms of these simulation tools are compared in various aspects including supported OS, simulation engines, traffic generators, and GUI tools. NCTUns supports real-life application program simulation due to the kernel re-entering technology used in the simulation engine. Ns-2, QualNet, and OPNET Modeler need to modify or re-implement the real-life applications. Thus the performance of real-life applications cannot be exactly evaluated in ns-2, QualNet, and OPNET Modeler. On the other hand, NCTUns can execute the real-life applications without any modification and thus can accurately evaluate their performance under various network conditions.

The paper compares the WiMAX-related functions implemented in these simulation tools. Furthermore, the paper reports the performance of NCTUns with Ethernet network simulation and WiMAX network simulation. In Ethernet network simulation, our study indicates that NCTUns requires much less CPU execution time and slightly more memory usage than the commercial tools OPNET Modeler 12.0 in Ethernet simulation. However, OPNET Modeler provides more functions than NCTUns in terms of WiMAX-related parameter configurations. For example, the OPNET Modeler supports various scheduling algorithms in the MAC layer, while NCTUns only supports Round-Robins algorithm. The OPNET Modeler provides many QoS mechanisms, while NCTUns provides UGS mechanism. In WiMAX network simulation, NCTUns requires a little more CPU execution time and memory usage due to the overhead resulted from the real-life application supports in NCTUns. As a final remark, NCTUns is a free software and the users can flexibly develop their own WiMAX functions on top of this free software.

6. ACKNOWLEDGMENTS

This work was sponsored in part by NSC 96-2752-E-009-005-PAE, NSC 95-2218-E-009-201-MY3, NSC 95-2221-E-009-024, NSC 95-2219-E-009-010, NSC 96-2219-E-009-019, Intel, ITRI/ NCTU Joint Research Center and MoE ATU.

7. REFERENCES

[1] The NCTUns tool. *available at http://nsl.csie.nctu.edu.tw/nctuns.html.*

[2] The Network Simulator - ns-2. *available at http://www.isi.edu/nsnam/ns.*

[3] The Object TCL language. *available at http://otcl-tclcl.sourceforge.net/otcl/.*

[4] The OPNET modeler. *available at http://www.opnet.com/.*

[5] The Parsec language. *available at http://pcl.cs.ucla.edu/projects/parsec/.*

[6] The POSIX standard. *available at http://standards.ieee.org/regauth/posix/.*

[7] The Protol-C language. *available at http://www.moselle.com/jsrs/protoc/protoc.html.*

[8] The QualNet software. *available at http://www.scalable-networks.com/.*

[9] The WiMAX forum. *available at http://www.wimaxforum.org/home/.*

[10] The WiMAX module for ns-2. *available at http://ndsl.csie.cgu.edu.tw/wimax_ns2.php.*

[11] IEEE standard for local and metropolitan area networks, part 16: Air interface for fixd broadband wireless acess systems. June 2004.

[12] H. Chao and Y. Lin. *WiMAX/MobileFi: Advanced Research and Technology, Chapter title: QoS Support in IEEE 802.16-Based Broadband Wireless Networks.* Auerbach Publications, New York, 2007.

[13] S. Wang, C. Chou, and C. Lin. The design and implementation of the nctuns network simulation engine. *Simulation Modelling Practice and Theory*, 15(1):57–81, January 2007.