WINSE: WiMAX NS-2 Extension

A. Sayenko
Research & Technology Platforms
Nokia Siemens Networks
Espoo, Finland
alexander.sayenko@nsn.com

O. Alanen, H. Martikainen, V. Tykhomyrov, O. Puchko, T. Hämäläinen
Telecommunication laboratory, MIT department
University of Jyväskylä, Finland
{olli.alanen,henrik.martikainen,vitykhom, olpuehko,timoh}@jyu.fi

ABSTRACT

IEEE 802.16 standard defines the wireless broadband technology called WiMAX. When compared to other wireless technologies, it introduces many interesting advantages at PHY, MAC, and QoS layers. Heavy simulations are needed to study IEEE 802.16 performance and propose further enhancements to this standard. Link level simulations are not always sufficient, while system level simulators are not always accurate to capture MAC and transport protocol details. We implemented a 802.16 extension for the NS-2 network simulator. It includes upper PHY modeling, almost all the features of the 802.16 MAC layer, as well as the QoS framework. This article describes the implemented features, simulation methodology, and shares our experience that can be used with other NS-2 modules. An overview of research papers, where this implementation was used, is given.

Keywords

IEEE 802.16 WiMAX, NS-2

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*

1. INTRODUCTION

IEEE 802.16, called WiMAX, is a standard for the wireless broadband access network [2, 3] that can provide a high-speed wireless access to home and business subscribers. It can ensure diverse Quality-of-Service (QoS) requirements which makes IEEE 802.16 a scalable platform for many services [24]. The core components of a 802.16 system are a subscriber station (SS) and a base station (BS). The BS and one or more SSs can form a cell with a point-to-multipoint (PMP) structure. In this case, the BS controls the activity within a cell, resource allocations to achieve QoS and admission based on the network security mechanisms. To support bidirectional communications, IEEE 802.16 supports both time division duplex (TDD) and frequency division duplex (FDD) operations.

As any other wireless technology, IEEE 802.16 emerged from a huge number of technical contributions. This technology continues to evolve through technical corrections, best practices, and more radical proposals. All these changes are supported by heavy simulations performed by researchers and engineers at various architectural levels. It is quite difficult, or even impossible, to accept a contribution if it is not based on reliable simulation results.

Traditionally, wireless network simulations are done with two different types of simulators. The first type of them is the link level simulators, in which a link between a base station and a subscriber station is modeled with a great level of details. The output from these simulators is usually bit, block and packet error probability under different parameters and conditions.

The second large group of simulators, which is widely adopted by industry, is system level simulators. They model a number of geographical cells where base stations provide service to subscriber stations. System simulators do not usually model all the link level aspects, but rather rely on the results obtained from the link level simulations. Nevertheless, their level of details include sub-carriers and a particular permutation type. System simulators are classified further into static and dynamic ones. Static simulators assume that the subscriber stations do not move and the whole system is modeled with several "snapshots" of time, where SS positions are randomly distributed over the simulation area. In the dynamic system simulations, SSs can move over the simulation area performing network entries and handovers between the BSs. Therefore, the dynamic system simulators capture simulation results as a function of time. Their only disadvantage is significant complexity and long simulation times.

Third option for wireless simulations is a packet level simulator, such as NS-2 [39]. When compared to the dynamic system simulators, packet level simulators are very similar in terms of provided features. However, protocol stacks and application behavior are modeled usually much more accurately. Similarly to the dynamic simulators, they account for many PHY aspects through abstractions and interfaces to link level simulators. Packet level simulators also allows for simulating the access service networks because it is possible to define a network topology where base stations, routers, gateways, and clients send or receive data. This makes it possible to obtain true end-to-end simulation results.

The rest of this paper is organized as follows. Section 2

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5599
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5599

gives a brief overview of other NS-2 802.16 modules. Section 3 provides details of our 802.16 extension. We do not delve into specific implementation details, but rather elaborate on simulation methodology and trade-off between complexity and simulation time. Section 4 gives an overview of past and ongoing research topics studied with our 802.16 module. Finally, section 5 concludes the paper.

2. PREVIOUS WORK

In this section we give a brief overview of existent 802.16 modules for NS-2 that we are aware of. We refrain from any evaluation – our aim is to provide an insight on typical features that other wireless broadband packet-level simulators have or do not support. A more practical comparison and analysis of different 802.16 modules can be found in [12].

2.1 NIST module

NIST module is definitely one of the first 802.16 extensions for the NS-2 simulator. Even though the project goal aimed merely at studying the inter-network handovers, its result was also 802.16 MAC, handover, and scheduling extensions. The list of supported features as well as the general description is given in [30]. As a small summary, there is almost no proper PHY with a correct error generation. There is a OFDM PHY emulation, whereas WiMAX is based on OFDMa PHY. Absence of the ARQ mechanism makes it complicated to deploy the error model. The MAC level lacks several important features, such as packing.

2.2 NDSL module

This module is a result of a joint work between Chan Gung University and Institute for Information Industry. The module description and features are presented in [13]. Somewhat similarly to the NIST module, it focuses mostly MAC leaving the PHY level unattended. However, the MAC level operational parameters correspond to the OFDMa PHY. In addition, the MAC level implementation includes both fragmentation and packing; special attention to the management messages and network entry procedure is given. ARQ and HARQ retransmission mechanisms are not implemented. Despite a good set of features, further development of this module has stopped.

2.3 WiMAX Forum module

The WiMAX Forum has been developing its NS-2 802.16 extension that emerged from the NIST module. Since this module is available only for the WiMAX Forum members, only a brief overview will be given. Unlike a module from NIST, it focuses OFDMa PHY and tends to emulate the PHY behavior at the sub-carrier level. However, several important PHY features, such as channel reporting and link adaptation, were not introduced. The MAC level lacks a full support for ARQ; the HARQ retransmission is not implemented at all. Unfortunately, the WiMAX Forum announced that the development process will stop and the future of this module is not clear. Nevertheless, this module is very important in a sense that this was one of the first attempts to introduce PHY at the sub-carrier level granularity.

2.4 Other modules

There is a 802.16 extension for NS-2 from the Eurecom Institute. This is a relatively new module, features of which are described in [27]. Even though authors present it as a novel module with integrated QoS architecture, it does not differ in principle from the NIST implementation. The PHY model is not revised at all and the MAC timing works in accordance with OFDM PHY parameters.

There is also a module from KAIST university [20] and 802.16 extension for the NS-2 MIRACLE framework [10].

2.5 Pisa university 802.16d Mesh module

This module implements an alternative to PMP 802.16 Mesh mode. However, IEEE 802.16 working group discontinued 802.16 Mesh mode that is removed completely from the IEEE 802.16 evolution [4]. Absence of a standardization and industry support makes it quite complicated to compete with other ad-hoc technologies, such as 802.11s.

3. WINSE

3.1 Overview

WINSE is a WiMAX extensions for the NS-2 simulator. It was started as a small student project and then evolved into a powerfull simulation tool that now several companies use to study the MAC and QoS in the 802.16 system. Table 1 gives a short overview of features supported in WINSE.

Table 1: Features supported by WINSE.

PHY
OFDM and OFDMa PHY
FEC blocks
HARQ: Type I, UL ACK channel
Channel reports: REP-RSP and CQICH
Link adaptation
MAC
DL broadcast messages: DL-MAP, UL-MAP, DCD, UCD
Compressed MAP, sub-MAPs
Connections: basic management & transport
PDU construction
Fragmentation & packing
Bandwidth requests: standalone & piggy-backed
ARQ: blocks, feedbacks, timers, transmission window
Uplink contention: OFDM and CDMA-based for OFDMa
Network entry
Handover
Sleep mode
QoS & scheduling
UGS, ertPS, rtPS, nrtPS, BE
BS scheduler
SS uplink scheduler
Access service network
ASN-GW
R4, R6, and R8 interfaces

3.2 Core principles

Before delving into the technical details, it is worth mentioning core principles behind our 802.16 module.

- Preserving existent framework. While introducing the 802.16 extensions, we do not change the NS-2 core classes, but rather introduce new functionality by creating new ones. As considered in many papers, the

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5599
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5599

NS-2 core lacks many features and sometime suffers from a bad internal design. However, our approach is that researchers should be able to combine different modules on a common simulation platform to study more complex scenarios. Radical changes should be addressed by major revisions, e.g., NS-3.

- C++ modularity. Emerged implicitly from the previous principle, there should be a class hierarchy with well defined class responsibilities. Fig. 1 shows the UML diagram with the top-level classes that constitute a core of our 802.16 module. Each functional block is contained in an independent C++ class that allows for further virtualization and abstraction.
- OTcl modularity. The most critical and fundamental C++ classes are mapped to the correspondent OTcl classes. As can be seen from Fig. 1, all the major classes are derived, either explicitly or implicitly, from **TclObject**. It allows a script designer to change easily parameters and switch between different modules just by selecting their OTcl class names.
- Balance between the PHY and MAC features. The power of the NS-2 simulator is in transport and application level. It is not reasonable to go into the deep PHY modeling as it will increase significantly the computational burden. We select carefully PHY features to model and ways to model them to ensure that NS-2 does not turn into the link-level simulator.

3.3 PHY Layer

3.3.1 PHY abstraction

A properly designed PHY level must introduce an abstraction general enough to hide particular PHY details from other architectural components, such as MAC and scheduler. All the 802.16 modules mentioned in section 2 have a PHY specific MAC implementation resulting in either OFDM or OFDMa 802.16 simulator. Even though the industry chose OFDMa PHY as a basis for WiMAX networks, there are a lot of OFDM devices. Furthermore, an upcoming 802.16m PHY also dictates a need in supporting simultaneously several PHYs.

Based on our experience, it is possible to provide such an abstraction if PHY exposes at least the following parameters:

- duplexing mode
- OFDM symbol duration
- zone parameters (number of OFDM symbols in a single slot and number of channels in one symbol)
- TTG and RTG gaps
- effective slot size for each modulation and coding scheme (MCS)

Having this information, the MAC level can work without knowing particular PHY details. This abstraction is also important for the scheduler. As an example, our implementation has a common MAC level scheduler that works correctly on top of both OFDM and OFDMa. As can be seen

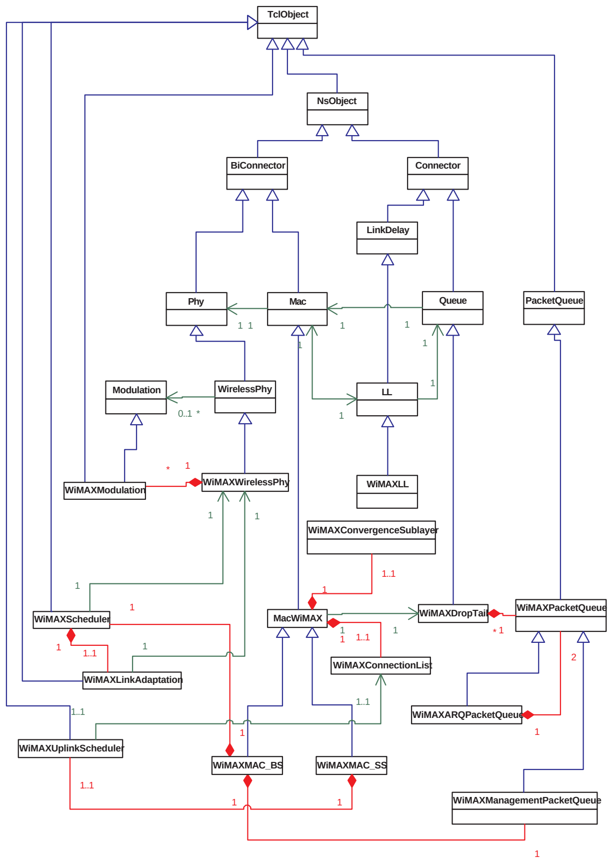

Figure 1: UML diagram for WINSE core classes.

from Fig. 1, all the core components have an association to the **WiMAXWirelessPhy** class that abstracts particular PHY sub-classes.

3.3.2 Effective SINR

One of the most critical issues for a packet level simulator is how obtain a valid effective signal-to-interference noise ratio (SINR). As we will present later, once the packet level simulator has an effective SINR, it can model accurately upper PHY functionality, such as PDU errors, HARQ operations, channel reports, link adaptation etc.

Many NS-2 researchers and developers assume that the NS-2 wireless framework already addresses all the necessary PHY aspects and 802.16 PHY is just a question of changing existent parameters or choosing slightly different models [11]. Unfortunately, NS-2 is far even from capturing all the necessary 802.11 PHY aspects [14]. Its logical design corresponds to a single-carrier case, to which path loss, and optionally antenna and fading models are applied. In reality, both 802.11 [1] and 802.16 PHY rely upon the OFDM technology with multiple sub-carriers. It means that each sub-carrier can experience a different path loss, fading, interference and so on. On the one hand, in 802.11 and 802.16 OFDM PHY we can assume that all the sub-carriers have exactly the same behavior thus working with a single carrier that will represent an effective SINR. To some extent, it is a valid approach for 802.16 OFDM where a slot al-

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5599
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5599

ways maps to all the sub-carriers in the OFDM symbol; the same holds for 802.11. On the other hand, sub-carriers may have quite different SINR values due to partially overlapping bandwidth (typical 802.11 case), a small guard band or interfering cells (802.16 case). Another important PHY aspect that usually NS-2 researchers do not account for is the UL sub-channelization gain. Since an SS transmission power must be distributed evenly between all the UL allocation sub-carriers, the size of the allocation has an impact on received signal strength.

Practically, it is not very complicated to introduce sub-carriers in NS-2. They can be modeled quite easily over the existent **Channel** class without even changing the core framework. On top of that, one can add path loss, antenna pattern, shadowing/slow fading, fast fading, etc. All the related models and algorithms are well known and defined [6, 29]. The problem is that we need the effective SINR that is calculated based in individual sub-carrier SINR values. The more sub-carries we have, the more computational resources are needed. The 802.16 OFDM PHY has 256 sub-carriers; in 802.16 OFDMa, there can be up to 2048 sub-carriers, e.g., in 20 MHz channel. Furthermore, in OFDMa PHY, sub-carriers can be either adjacent or distributed over the whole bandwidth. The problem becomes even more computationally expensive if we start to model multiple cells. It is not reasonable to turn NS-2 into another link level simulator as its power in accurate modeling of higher layers. It is also worth mentioning that interference calculation involves heavy computations. Even system level simulators use simplifications here.

Based on these considerations, our current approach is to obtain effective SINR from trace files generated with dynamic system simulators. It allows for a good trade-off between computational complexity and accuracy. The implementation relies upon the NS-2 **Propagation** class, from which we derive a new class to hide trace file implementation details.

Nevertheless, we still explore possibilities to introduce a more advanced PHY model for NS-2 because the trace file approach has obvious limitations. The trace files are generated from a particular environment, particular node locations, particular traffic mixes, loads and so on. As an example, it is impossible to study the channel aware scheduling with SINR taken from the trace files.

3.3.3 Error generation

The error generation model is responsible for answering a question whether a received protocol data unit (PDU) is erroneous or not. To model it, we account for the way 802.16 encodes and sends data. Each data allocation, i.e., a burst, is an integer number of contiguous slots. On top of that, slots are grouped into the forward error correction (FEC) blocks, where the FEC block size depends on a particular MCS. A PDU can start and end on *any* byte within a data burst. All these layers are presented in Fig. 2. Thus, to model errors correctly, we have to map a received PDU to the FEC blocks it spans.

To accomplish correct PDU to FEC block mapping, we instruct the PHY layer to build a FEC block list whenever a new burst starts. Then, inside the **sendUp()** function, we track the number of received bytes so that whenever a new PDU arrives, we can determine the starting and ending FEC block. Once the list of FEC blocks, to which the received

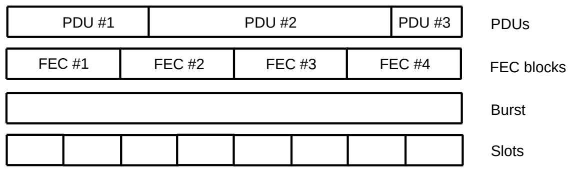

Figure 2: Slots, burst, FEC blocks, and PDUs

PDU maps, is known, it is passed to the error generation module. Each FEC block carries information on its size and SINR. The error generator applies the following formula

$$\mathbf{E} = 1 - \prod_i (1 - E_i), \tag{1}$$

where E_i is an individual FEC block error probability determined based on the FEC BLER curves, as Fig. 3 shows.

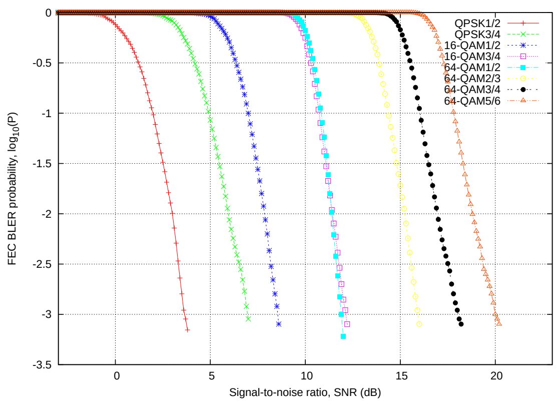

Figure 3: CTC FEC BLER curves (only the largest FEC block sizes).

It is important to note that FEC BLER curves presented in Fig. 3 are not hardcoded but rather specified as the OTcl level. It allows to switch easily between different link level simulation results and coding schemes, e.g., CC and CTC.

3.3.4 HARQ

The HARQ Type I, i.e., Chase Combining (CC), implementation and modeling follows the 802.16 simulation methodology [6]. Every time a new HARQ retransmission is made, the FEC block SINR from all the previous (re)transmissions is summed and submitted to the error generation module considered earlier. The HARQ Type II, Incremental Redundancy (IR), is more complicated to model with NS-2 without going into coding and decoding details. One approach is to model it on top of Type I as positive or negative gain based on retransmissions' SINR [18, 15, 19]. Another approach is to have different FEC BLER curves for the first retransmission, second one, and so on [21]. Anyway, HARQ IR mode is not so important for WiMAX networks as it is not mandated by the system profile [7].

In the case of UL HARQ transmission, the BS always knows the burst reception status. In the case of DL HARQ burst, an SS reports back the HARQ status via the HARQ ACK channel. Once the BS scheduler knows the burst reception status, it can decide whether to schedule a HARQ retransmission or continue will allocating data on next free HARQ channels.

Our HARQ implementation fully conforms to the 802.16 specification in a sense of supported and adjustable parameters. It is possible to specify the maximum number of HARQ channels (16 by default), maximum number of HARQ retransmissions (4 by default), HARQ buffer mode (shared by default), UL HARQ ACK delay (1 frame by default).

3.3.5 *Repetition factors*

The repetition factor is modeled similar to HARQ Type I. The received packet power is just multiplied by the repetition factor and then passed to the error generation model.

3.3.6 *Channel reports & link adaptation*

When the BS receives data, it can always estimate the channel to switch to a more suitable UL MCS in the next frame. When an SS receives data, it has to estimate the channel and report it back to the BS so that the BS link adaptation can also choose a suitable DL MCS. We support two reporting mechanisms: REP-RSP messages and CQICH channel. While the former is available in both OFDM and OFDMa PHY, it is less reliable because REP-RSP is an ordinary MAC level management message that can be dropped easily. CQICH channel, which is defined only for OFDMa PHY, provides a more robust way to report channel status.

The implementation of the REP-RSP message is quite straightforward. 6-bit CQICH messages are modeled with a special NS-2 packet type, payload of which carries the necessary information.

Once the BS scheduler link adaptation module (see Fig. 1) has information on all the SSs DL and UL SINRs, it can adjust MCS to achieve the target FEC BLER for each connection. Refer to section 4.7 for more information.

3.4 MAC Layer

3.4.1 *Queue system*

The general structure of the queue system is presented in Fig. 4. From the BS point of view, the air interface is a bottleneck which creates a need to buffer DL packets. Similarly, an SS needs queues to buffer UL packets. To differentiate between service flows and ensure QoS, each connection is allocated a separate queue. In addition, the BS keeps so-called UL virtual queues maintained through the bandwidth requests transmitted by SSs.

Each transport connection is equipped with several internal sub-queues where it keeps initial transmissions, retransmissions, and ARQ feedbacks. Internal priority queuing (PQ) scheduler ensures that first a connection will send ARQ feedbacks, then retransmissions, and only then normal PDUs. By the default, a sub-queue for initial transmissions relies upon a simple drop tail method. Alternatively, some Active Queue Managamenet (AQM) mechanism can be applied there. See section 4.3 for more details.

At the BS side, there is also a special DL queue where the scheduler puts generated DL-MAP, UL-MAP, DCD, and UCD messages. Besides, certain management messages, such as MOB_TRF-IND, are also placed into this queue as they are designated to all the SS.

3.4.2 *MAC level timers*

It is worth mentioning briefly a scalable design for the

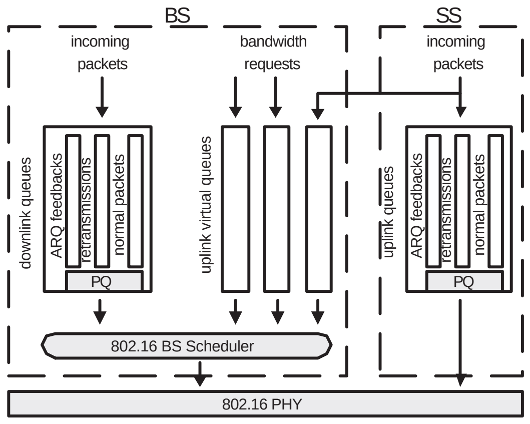

Figure 4: The queue system.

MAC level timers that enables our module to work in both the TDD and FDD modes. Fig. 5 shows that there is a top level timer that elapses whenever a new frame should start. The second level is a burst timer that ensures a transition from one burst to another. Whenever a new burst starts, the MAC level prepares PHY and resets certain MAC level state variables. The third level is a PDU timer that elapses whenever data transmission (or reception) ends. There is also a gap timer that simulates various transition gaps, such as TTG and RTG.

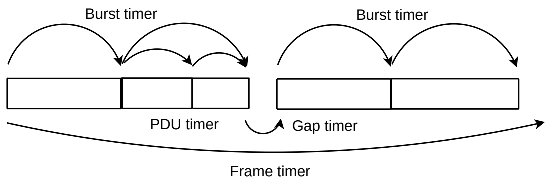

Figure 5: MAC level timers.

Fig. 5 shows a simplified timer functioning for the TDD mode where DL bursts are followed by the UL bursts. In the FDD mode, two burst timers function in parallel to support simultaneous data transmission and reception.

3.4.3 *Packet header suppression*

The packet header suppression is implemented in a very simple way. A script designer just specifies a constant suppression size. Whenever a packet is placed to the queue, its size is decremented to emulate suppression. Then, when a packet is passed to LL, the same constant size is added back.

3.4.4 *Packing & fragmentation*

While implementing packing and fragmentation, we faced biggest problems with the NS-2 framework. There is no internal support for such a simple operation as fragmentation. This functionality must be implemented on top of NS-2 **Packet** class. Even worse thing is that a bad C++ design of the **Packet** class prevents from applying virtualization and abstraction. Packing caused even bigger problems as it implies that several NS-2 packets, or even packet fragments, are transmitted as one entity. Our solution for packing is to

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5599
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5599

have a NS-2 packet with a special payload that keeps all the embedded packets or packet fragments.

3.4.5 *ARQ mechanism*

The ARQ mechanism does not contain any NS-2 specific features. Since it works on top of MAC, it is a straightforward implementation from the 802.16 specification. Once the MAC level detects missing ARQ blocks, it activates the ARQ mechanism and informs a sender to retransmit data. ARQ timers take care of situations when ARQ feedbacks are lost constantly or when retransmission attempts expire.

It makes sense to mention only one important optimization that we use in the ARQ implementation. Instead of creating and running ARQ timers for each ARQ block that a PDU has, we associate them with a whole PDU. This is an approach that also certain WiMAX products use. Of course, a special take care must be taken when later a retransmitted PDU is fragmented or packed.

The ARQ mechanism implementation supports the following features: ARQ blocks, ARQ block rearrangement, ARQ feedbacks (standalone and piggy-backed), ARQ window, ARQ timers (retry, block lifetime, Rx purge), ARQ discard.

3.4.6 *Contention resolution*

Depending on the underlying PHY, the 802.16 contention resolution works differently. Even though the top level back-off mechanism is identical to both OFDM and OFDMa PHY, the former just sends a 6-byte PDU header with the bandwidth request size. In OFDMa PHY, there are 256 144-bit pseudo-orthogonal CDMA codes. An SS sends first a CDMA contention code; once the BS detects the code, it allocates a special uplink CDMA allocation where an SS can send a bandwidth request. All these differences are abstracted through the MAC level contention resolution class.

The collision in OFDM PHY is detected and handled quite easily – if the BS detects transmission while the previous one has not ended yet, then all the packets are dropped. The OFDMa PHY uplink contention is trickier to model in NS-2 because it is a tradeoff between accuracy and computational complexity. When several SSs transmit simultaneously CDMA codes, the BS tries to detect each of the transmitted codes. Several approaches for modeling a CDMA receiver in NS-2 are available.

1. Optimistic. A CDMA code is modeled with a special NS-2 packet type, where the packet payload carries just the CDMA code index. Since CDMA codes are pseudo-orthogonal, we can assume that there is a high probability that all the transmitted codes are detected. The only case when a code collision occurs is when two or more identical codes are sent. Thus, the BS CDMA receiver just analyzes CDMA code indexes to decide whether they are detected or not.

2. Simple. This receiver is similar to the previous one with one addition – it tries to account for a inter-code interference based on the link simulation or other simulation results [17]. Even if two or more received codes have different indexes, there is a chance that a code is not detected.

3. Single user. This approach tries to model the way the CDMA codes are transmitted and detected by the CDMA receiver. The NS-2 CDMA packet payload carries 144 bits. If there are several codes transmitted during the same transmission opportunity, then the received bit sequences are summed to obtain a so-called interference pattern. Since each bit is BPSK modulated, there is a non-zero probability that its value changes during transmission (see Fig. 6). Thus, a receiver models it by applying the BPSK curve to each bit. Then, the BS CDMA receiver tries to detect individual codes by applying the dot product operation.

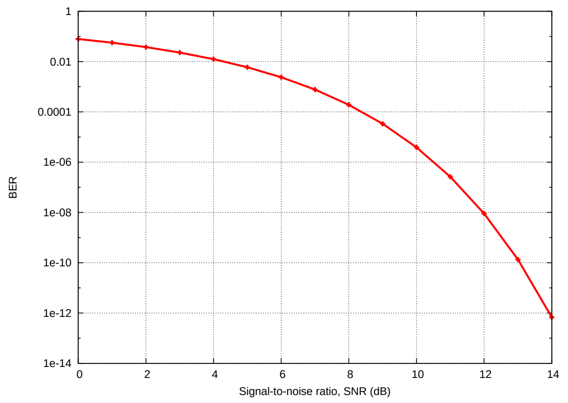

Figure 6: BPSK bit error rate.

Depending the simulation accuracy and speed, an appropriate CDMA receiver can be chosen. As an example, an accurate VoIP delay analysis, where uplink ertPS connections rely upon the contention, definitely requires a complicated CDMA receiver. On the contrary, a simple CDMA receiver suffices for a basic analysis of the TCP performance over 802.16.

3.4.7 *Handover*

Handover is a part of the basic MAC functionality and includes a support for all the necessary management messages to initiate and control the handover process. Our implementation is limited in a sense that there is no module to initiate the handover process – it must be initiated manually from the Tcl script; neither do we support scanning. Future WINSE versions may address this functionality.

3.5 LL layer

The LL layer has the least amount of extensions when compared to MAC or PHY. Following the general principles, there is a derived class that changes behavior of the virtual **recv()** function. Firstly, we disable Address Resolution Protocol (ARP) because it does not have much sense in the 802.16 PMP mode. Secondly, the LL layer has to stamp a packet with a correct CID so that the queue system can place it into a correct queue. Practically, the NS-2 LL layer performs functions of the 802.16 convergence sublayer. This functionality is identical for an SS and BS as both of them have to classify incoming packets. Current implementation classifies packets based on the following parameters: a) source address, b) destination address, and c) flow ID. The reason we also account for flow ID is that we have to differentiate somehow between incoming packets that belong to different applications from the same node. If they belong to different application types, e.g., VoIP and BE, then they should be placed into different connection queues to obtain an appropriate treatment. On the other hand, by not specifying the flow ID we will put all the incoming packets into one connection queue which is also a valid case. A script designer decides which option to use.

3.6 QoS and scheduling

The internal QoS architecture of our 802.16 module does not differ significantly from QoS architectures presented in other papers and 802.16 implementations [16, 13]. Thus, for the sake of brevity, we focus only on distinctive features we adopted in our module.

3.6.1 BS scheduler

We omit the description of the BS scheduler details as more information is given in section 4.1. However, we mention briefly that to study different schedulers in 802.16, we introduced a common interface between the BS MAC and the BS scheduler, as Table 2 presents.

Table 2: BS scheduler interface.

Input	Output
DL queue sizes	DL burst list
UL queue sizes	UL burst list
HARQ ACKs	DL-MAP, UL-MAP, DCD, UCD

The input parameters are the status of DL and UL queues that are supported and managed by the MAC layer. In addition, we also pass HARQ ACKs because they arrive to BS via the MAC level. The result of the scheduling decision is two lists with DL and UL bursts for the BS MAC. The scheduler also constructs the DL-MAP, UL-MAP, DCD, and UCD messages that the BS MAC transmits later to all the SSs.

3.6.2 SS uplink scheduler

The SS uplink scheduler is as important as the BS scheduler. The reason is that once the BS makes an UL allocation, it is per a whole SS, not per an individual connection. Then, it is the SS responsibility to partition this UL allocation between multiple transport connections, if any. Such a solution aims at reducing the signaling overhead and allowing an SS to gain a better control on the allocation size, which can start and end at any byte within the burst. Fig. 7 shows an example of how an UL burst can be shared between several connection types.

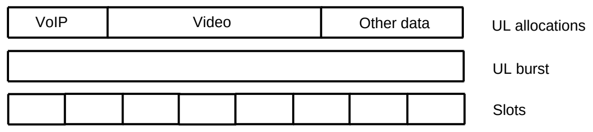

Figure 7: UL burst partitioning.

The default uplink scheduler in our implementation is PQ. Its simple yet efficient design allows for supporting triple-play services at mobile terminals. In other words, first the uplink scheduler allocates space for the management data, if any, then UGS, then ertPS (e.g., VoIP), then rtPS (e.g., video), and then nrtPS (e.g., Web) with BE.

Of course, the PQ scheduler cannot address more complicated and general cases when, as an example, there are several nrtPS connections with diverse QoS requirements, such as the 802.16 CPE that serves a small local network. To experiment with different uplink schedulers, we developed a common interface that allows to abstract from a particular allocation algorithm (see Table 3). Input parameters are time, UL allocation size, and the queue sizes. Output information is a list with UL allocation sizes that tell how many bytes each connection may occupy. Then, it is the MAC level responsibility to enforce this decision.

Table 3: Uplink scheduler interface.

Input	Output
Time	UL allocation list
UL allocation size	
UL queue sizes	

3.7 Routing

While working in the PMP mode, packets are always exchanged between SSs and the BS belonging to the same cell – there is no way an SS can communicate to the other one bypassing the BS. As a result, there is no need in the ad-hoc routing protocol because all the forwarding information becomes available during a connection set up. To reflect this fact, we rely upon the NOAH module that just disables ad-hoc routing.

3.8 Tracing

Following one of the core principles, we rely upon the trace format provided by NS-2 framework. We do not change the trace file format as it will break existent scripts or start to conflict with other extensions. However, there is 802.16 specific information that one would like to see in the trace file for further analysis and testing.

The NS-2 wireless trace file format defines four `-Mxxx` fields, content of which is specific to a particular MAC level. We redefine them to include 802.16 specific information, as Table 4 presents. The `-Mt` field always have the same value, 802.16, which allows to differentiate from other MAC types. The `-Mc` field keeps the CID value. It helps to differentiate between several connections belonging to same SS or just track down a particular connection. The `-Mm` field specifies the management message type if a transmitted PDU belongs to the management connection. Having the management message type in the trace file, one can gather plenty of important information. As an example, network entry delay (time between the first RNG-REQ and the last REG-RSP messages), connection setup delay (time between the first DSA-REQ and the last DSA-ACK messages), handover delay (time between the first MOB_MSHO-REQ and the last REG-RSP messages), and so on. The `-Mb` field specifies the bandwidth request size of a standalone or piggy-backed bandwidth request.

In addition to the new `-Mxxx` fields, we add a special ”c” entry to the trace file. It specifies when an SS takes part in the uplink contention resolution. The reason we need this entry is that when an SS starts the uplink contention, it may defer for a number of frames due to the backoff start value. As a result, the first bandwidth request is sent later when the actual uplink contention has begun. Thus, the ”c” entry

Table 4: New trace fields.

Field	Description
-Mt	802.16
-Mc	CID
-Mm	management message type
-Mb	bandwidth request size

helps to measure accurately the medium access delay. Since an SS performs the uplink contention on behalf of all the connections it has, the format of this entry is very simple, as Fig. 8 presents. There is time when a contention starts, SS node ID that originates the contention, and the BS node ID to which the contention request is sent.

```
c -t <time> -Hs <ss_id> -Hd <bs_id>
```

Figure 8: Format of the contention entry.

There are many reasons why a packet can be dropped inside the MAC layer. We rely upon the existent NS-2 drop reasons to put appropriate information to the trace file.

Table 5: Drop reasons.

Drop reason	Description
IFQ	The queue is full
ERR	Packet error
COL	Uplink contention collision (OFDM only)
RET	Contention attempts exceeded
NRTE	No classification rule

3.9 Access network

Along with the 802.16 radio interfaces, the performance of the WiMAX network is affected by the wired part. In fact, wired network components, such as access service network gateway (ASN-GW) and core switching node (CSN) (see Fig. 9), may play a crucial role during network entry, connection setup, and handover processes because the BS contacts them at various stages [8]. Failing to model the access network, one can obtain too optimistic results. Apart from time needed to send a signaling message and wait for a response, there are also delays that may come from the user data preempting management traffic inside the access network.

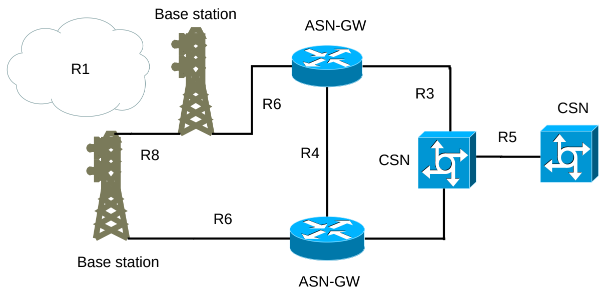

Figure 9: WiMAX access network components and interfaces.

Access network modeling combined with 802.16 PHY and MAC is a perfect task for a simulator, such as NS-2, where wired networking has been present for a long time. Besides, NS-2 provides a good framework to develop new protocols, in particular signaling ones.

We already developed an extension to the WINSE module that aims at adding a support for the 802.16 access network entities to the NS-2 simulator. Our primary goal is to support R4, R6, and R8 interfaces to obtain more accurate results for network entry and handover procedures. The description of the module with simulation results is given [26]

4. RESEARCH DONE WITH WINSE

In this section we give a brief overview of past and ongoing research topics where the WINSE is used.

4.1 Scheduling & resource allocation

Scheduling and resource allocation was the first research topic studied with the WINSE extension [35, 34]. It was shown that the classical fair resource allocation is applicable but is too complicated for the 802.16 networks, where the basic allocation unit is a slot of a fixed size and duration. As a result, a simpler yet efficient algorithm is possible that is based conceptually on the round-robin approach. At the same time, there were proposed extensions that allowed for accounting for the 802.16 service class, QoS parameters, the UL bandwidth request or DL queue size. The simulation results showed that the proposed scheme ensures all the QoS requirements, protects service flows, and ensures fair resource allocation across the BE connections. All our subsequent research papers relied upon the proposed scheduler that was adopted easily for the OFDMa PHY. Its simple but scalable nature also allowed to introduce an extension for ARQ [33] and a support for the HARQ scheduler.

Due to a highly modular internal architecture, other schedulers, such as proportional fair, were studied with the WINSE module [23].

4.2 Uplink contention performance

Even though the 802.16 system is based on the DAMA concept, there is still an uplink contention created by connections that are either not polled regularly, e.g., nrtPS, or are not polled at all, such as BE. The ertPS class can also take part in the uplink contention if so allowed by the BS. In [32], we studied the 802.16 uplink contention resolution and proposed a scheme on how to adapt dynamically backoff start/end parameters, and the number of the contention transmission opportunities. Furthermore, we showed in [31] that by adjusting dynamically the uplink contention parameters one can achieve a good tradeoff between resource utilization and delay requirements of the VoIP connections.

Another 802.16 feature to achieve the tradeoff between delay guarantees and resource utilization is multicast polling. In [9], a study was done to research the applicability of it with VoIP connections. It was shown that the multicast polling can be used to provide a maximum delay guarantee and even several separate delay limits for separate connection types.

Our recent paper studied the OFDMa PHY CDMA contention code performance and how they can be optimized for future broadband wireless systems [17].

4.3 AQM

From the SS and BS point of view, the 802.16 network is a bottleneck because it is highly anticipated that a wired connection behind BS (or SS working as a gateway) will be always higher than the maximum achievable throughput on the air interface. Thus, the queue sizes tend to grow especially when spectrum efficiency declines. In [22], we studied AQM mechanisms that it is possible to apply to 802.16 to reduce queuing delays. The results showed that being applied to the BS DL queues, the AQM mechanism is capable of reducing TCP delays.

4.4 Optimal PDU Size

While transmitting data, incoming packets, i.e., SDUs, are fragmented or packed into PDUs, size of which is not governed by the specification. On the one hand, the propability of errornous MAC PDU increases when the PDU size grows. On the other hand, a small PDU has a larger overhead. In [25], we studied the optimal MAC level PDU size in different channel conditions. We showed that it is possible to find the optimal PDU size if we know the channel conditions. Because 802.16 system has an advanced link adaptation mechanism, the error probability is known and the optimal PDU size can be selected. It was also shown that if the error probability is unknown, then it is better to rely on smaller PDU sizes of around 100-120 bytes.

4.5 ARQ

The ARQ retransmission mechanism is available in all the major PHYs of the 802.16 system and plays a key role in improving the system performance, especially the application level throughput. In [37], we studied the properties of the 802.16 ARQ mechanism and proposed solutions on how to choose an ARQ feedback type, prioritize feedbacks, retransmissions, and normal PDUs. We showed the importance of the ARQ block rearrangement and a correctly set up ARQ transmission window. In [38], we analyzed the impact of the ARQ feedback intensity in the case when ARQ works standalone and on top of HARQ.

4.6 ARQ & HARQ performance

Along with ARQ, the 802.16 OFDMa PHY provides a possibility to run the HARQ retransmission mechanism. In [36], we made a preliminary comparison of these two mechanisms and also studied ARQ on top of HARQ. Even though, as expected, HARQ outperforms ARQ in most cases due to a retransmission gain, there are cases when ARQ provides a better performance due to less signaling information.

4.7 Link Adaptation

In [28], we have analyzed the link adaptation model and MCS transition thresholds for the IEEE 802.16 base station. We have shown that the optimal transition threshold for the ARQ connections is between 10^{-2} and $10^{-2.5}$, while for the HARQ enabled connections it is from $10^{-1.5}$ to 10^{-2}. It fully conforms to the theoretical expectations that HARQ should outperform ARQ due to the retransmission gain. An interesting outcome of the paper is that the optimal thresholds depend on the number of data bursts per a frame. It requires a coordinated functioning between the BS link adaptation model and the scheduler.

5. CONCLUSIONS

The paper has presented the 802.16 module for the NS-2 simulator, called WINSE. We introduced a support for the upper PHY level, MAC, QoS and scheduling, as well as the basic support for the access service network. Our implementation has proved that NS-2 can be used to model complicated wireless broadband technologies, such as IEEE 802.16. We used the WINSE module in many research papers where we studied the MAC and QoS aspects of the IEEE 802.16 system. A number of technical contributions were also submitted. Ongoing research topics are FDD, sub-maps, and a support 802.16j multi-hop relay networks [5].

Based on our experience, it is possible to state that a truly powerful and scalable NS-2 module should not aim at a particular solution, but rather provide as many abstractions as possible to allow for different alternative implementations without changing the main core. The presented 802.16 module is an example of how it is possible to abstract from a particular PHY, BS scheduler, SS uplink scheduler, contention resolution mechanism, etc. It creates unlimited possibilities for future enhancements, such as 802.16j and 802.16m.

Acknowledgments

We express our acknowledgments to the following people who helped during development, took part in testing and provided valuable comments on implementation and simulation techniques: Juha Karhula, Levente Sajo, Timo Joensuu, Vesa Hytönen, Jani Lakkakorpi, Jani Moilanen, Thomas Höhne, Lauri Kuuru, Yanqun Le, Wu Yi, Zhang Dongmei, Jan Suumäki, Michael Gundlach, Milton Mendieta, Alexey Dudkov, Nenad Veselenovic.

Special thanks for the WiMAX Forum Application Working Group members Tom Tofigh, Shayem Parekh, and Raj Jain for supporting the simulation methodology work item.

6. REFERENCES

[1] Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. IEEE Standard 802.11, 1999.

[2] Air interface for fixed broadband wireless access systems. IEEE Standard 802.16, Jun 2004.

[3] Air interface for fixed broadband wireless access systems - amendment for physical and medium access control layers for combined fixed and mobile operation in licensed bands. IEEE Standard 802.16e, Dec 2005.

[4] Air interface for broadband wireless access systems. IEEE Standard 802.16 (Rev2 D5), Jun 2008.

[5] Air interface for broadband wireless access systems: Multihop relay specification. IEEE Standard 802.16j (D6), Jul 2008.

[6] IEEE 802.16m evaluation methodology document (EMD). IEEE 802.16 Broadband Wireless Access Group, Mar 2008.

[7] WiMAX Forum Mobile System Profile, Release 1.0 Approved Specification, Apr 2008. Revision 1.6.1.

[8] WiMAX Forum Network Architecture, Sep 2008. Release 1, Version 1.3.0.

[9] O. Alanen. Multicast polling and efficient VoIP connections in IEEE 802.16 networks. In *The 10th ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pages 289–294, Oct 2007.

[10] N. Baldo, F. Maguolo, M. Miozzo, M. Rossi, and M. Zorzi. NS-2 MIRACLE: a modular framework for multi-technology and cross-layer support in network simulator 2. In *NS tools*, Oct 2007.

[11] L. Betancur, R. C. Hincapie, and R. Bustamante. WiMAX channel – PHY model in network simulator 2. In *Workshop on NS-2*, Oct 2006.

[12] T. Bohnert, J. Jakubiak, M. Katz, Y. Koucheryavy, E. Monteiro, and E. Borcoci. On evaluating a WiMAX access network for isolated research and data networks using NS-2. In *7th International Conference on Next Generation Teletraffic and Wired/Wireless Advanced Networking*, pages 133–147, Sep 2007.

[13] J. Chen, C.-C. Wang, F. C.-D. Tsai, C.-W. Chang, S.-S. Liu, J. Guo, W.-J. Lien, J.-H. Sum, and C.-H. Hung. The design and implementation of WiMAX module for NS-2 simulator. In *Workshop on NS-2*, Oct 2006.

[14] Q. Chen, F. Schmidt-Eisenlohr, D. Jiang, M. Torrent-Moreno, L. Delgrossi, and H. Hartenstein. Overhaul of IEEE 802.11 modeling and simulation in NS-2. In *The 10th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Oct 2007.

[15] J.-F. Cheng. Coding performance of Hybrid ARQ schemes. *IEEE Transactions on Communication*, 54(6):1017–1029, Jun 2006.

[16] C. Cicconatti, L. Lenzini, and E. Mingozi. Quality of service support in IEEE 802.16 networks. *IEEE Networks*, 20(2):50–55, Mar/Apr 2006.

[17] A. Dudkov and A. Sayenko. Orthogonal bandwidth request codes for IEEE 802.16 networks. In *The 11th International Symposium on Wireless Personal Multimedia Communications*, Sep 2008.

[18] F. Frederiksen and T. Kolding. Performance and modeling of WCDMA/HSDPA transmission/H-ARQ schemes. In *IEEE Vehicular Technology Conference*, pages 472–476, Sep 2002.

[19] P. Frenger, S. Parkvall, and E. Dahlman. Performance comparison of HARQ with chase combining and incremental redundancy in HSDPA. In *IEEE Vehicular Technology Conference*, pages 1829–1833, Oct 2001.

[20] S. Kim and I. Yeom. IEEE 802.16 simulator. `http://cnlab.kaist.ac.kr/802.16/ieee802.16.html`.

[21] T. Kwon, H. Lee, S. Choi, J. Kim, D.-H. Cho, S. Cho, S. Yun, W.-H. Park, and K. Kim. Design and implementation of a simulator based on a cross-layer protocols between MAC and PHY layers in a WiBro compatible IEEE 802.16e OFDMa system. *IEEE Communications Magazine*, 43(12):136–146, Dec 2005.

[22] J. Lakkakorpi, A. Sayenko, J. Karhula, O. Alanen, and J. Moilanen. Active queue management for reducing downlink delays in WiMAX. In *Vehicular Technology Conference, 2007. VTC-2007 Fall. 2007 IEEE 66th*, pages 326 – 330, Sep 2007.

[23] J. Lakkakorpi, A. Sayenko, and J. Moilanen. Comparison of different scheduling algorithms for WiMAX base station: deficit round robin vs. proportional fair vs. weighted round robin. In *IEEE WCNC*, pages 1991–1996, Mar/Apr 2008.

[24] K. Lu, Y. Qian, H.-H. Chen, and S. Fu. WiMAX neworks: from access to service platform. *IEEE Network*, 22(3):38–45, May/Jun 2008.

[25] H. Martikainen, A. Sayenko, O. Alanen, and V. Tykhomyrov. Optimal MAC PDU size in IEEE 802.16. In *4th International Telecommunication Networking Workshop on QoS in Multiservice IP Networks*, pages 66–71, Feb 2008.

[26] M. Mendieta. Modelling of test specifics for multi-vendor WiMAX networks. Master's thesis, Mannheim University of Applied Sciences, 2008.

[27] I. C. Msadaa, F. Filali, and F. Kamoun. An 802.16 model for NS-2 simulator with an integrated QoS architecture. In *Simutools*, Mar 2008.

[28] A. Puchko, V. Tykhomyrov, and H. Martikainen. Link adaptation thresholds for the IEEE 802.16 base station. In *Workshop on NS-2 simulator*, Oct 2008.

[29] R. J. Punnoose, P. V. Nikitin, and D. D. Stancil. Efficient simulation of ricean fading within a packet simulator. In *IEEE Vehicular Technology Conference*, pages 764–767, Sep 2000.

[30] R. Rouil. The network simulator NS-2 NIST add-on: IEEE 802.16 model (MAC+PHY). Technical report, Jun 2007.

[31] A. Sayenko, O. Alanen, and T. Hämäläinen. Adaptive contention resolution for VoIP services in IEEE 802.16 networks. In *The 8th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, Jun 2007.

[32] A. Sayenko, O. Alanen, and T. Hämäläinen. Adaptive contention resolution parameters for the IEEE 802.16 networks. In *International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness*, Aug 2007.

[33] A. Sayenko, O. Alanen, and T. Hämäläinen. ARQ aware scheduling for the IEEE 802.16 base station. In *IEEE International Conference on Communication*, pages 2667–2673, May 2008.

[34] A. Sayenko, O. Alanen, and T. Hämäläinen. Scheduling solution for the IEEE 802.16 base station. *Computer Networks*, 52:96–115, 2008.

[35] A. Sayenko, O. Alanen, J. Karhula, and T. Hämäläinen. Ensuring the QoS requirements in 802.16 scheduling. In *The 9th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pages 108–117, Oct 2006.

[36] A. Sayenko, H. Martikainen, and A. Puchko. Performance comparison of HARQ and ARQ mechanisms in IEEE 802.16 networks. In *The 11th ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Oct 2008.

[37] V. Tykhomyrov, A. Sayenko, H. Martikainen, and O. Alanen. Analysis and performance evaluation of the IEEE 802.16 ARQ mechanism. *Journal of communications software and systems*, 4(1):29–40, Mar 2008.

[38] V. Tykhomyrov, A. Sayenko, H. Martikainen, O. Alanen, and T. Hämäläinen. On ARQ feedback intensity of the IEEE 802.16 ARQ mechanism. In *International Conference on Telecommunications*, Jun 2008.

[39] UCB/LBNL/VINT. Network simulator ns-2. `http://www.isi.edu/nsnam/ns`.