Implementation of Multi-service IEEE 802.11e Block Acknowledgement Policies

Orlando Cabral, Alberto Segarra, Fernando J. Velez, *IAENG*, *Member*

Abstract— **Optimisation of IEEE 802.11e MAC protocol performance can be performed by modifying several parameters left open in the standard, like buffer size and acknowledgement policies. In this work, an event driven simulator was developed, and a comparison of the results between the case of standalone services and several services supported simultaneously was addressed. With only a single service, higher values for the goodput are obtained for video and background (BK) traffics, while the lowest ones are found for the voice application. Besides, the number of supported users is higher for voice. With mixtures of traffic, when the number of station is small the goodput is lower for voice. However, for higher number of station, the lowest values of the goodput occur for background traffic. By using the Block Acknowledgement procedure, for video and background traffics in a single service situation, the capacity was improved in the case when the number of stations is equal or higher than 16 and 12, respectively. However, for lower values of the number of stations, the use of Block Acknowledgements leads to a slightly worst system performance. In a scenario with mixture of services the most advised block size is 12 (less delay in a highly loaded scenario). The total number of supported user increases from 30 to 35.**

Index Terms— **Simulation, IEEE 802.11e, Block ACK, EDCA.**

I. Introduction

Recent years have seen an immense growth in the popularity of wireless services and applications that require high throughput. To support such growth, standardization bodies such as the IEEE have formed task groups to investigate and standardize features providing increased quality of service and higher throughputs for IEEE 802.11. One such extension is the Block Acknowledgment (ACK) policy feature, included in the ratified IEEE 802.11e amendment [1, 2]. This feature improves system throughput by reducing the amount of overhead required by a station to acknowledge a burst of received traffic.

The Block ACK procedure improves system throughput results by reducing the amount of overhead required by a station to acknowledge a burst of received traffic [3, 4]. It acknowledges a block of packets by a single ACK, instead of using several ACKs, one for each packet. By doing this, the Arbitration Inter-frame Spacing (AIFS), the backoff counter, and the acknowledgement time periods are saved. The number of frame that can be transmitted within a block is called block size. It is limited but is not specified in the standard. In this paper, to find the most suitable block size we have tried several block sizes and several loaded scenarios with and without mixture of services.

Analytical frameworks to model Block ACK have been published [5, 6, 7] but the results are neither based on a realistic approach to the problem nor account for the achievable Quality of Service (QoS), because the use of several service classes with different priorities (the base for Enhanced Distributed Channel Access, EDCA) is not considered at all. The existing theoretical approaches [5, 6, 7] do not consider the hidden terminal problem, assume that the buffer is always full, and do not assume a multi-rate scenario.

In [5, 6], an analytical framework was presented to model an IEEE 802.11e ad-hoc network with the Block ACK procedure for a completely simplified scenario. Although the hidden/exposed terminal problem is one fundamental issue in WLANs most of the existing analytical models either assume it does not exist, or do not consider the EDCA features of IEEE 802.11e (or do not account for the delay or any other QoS metric) [5, 6]. Results presented in [5, 6] express the block size as a function of the goodput in saturation conditions. Results show that the block size should be as high as possible but this can be misleading when QoS is considered. In [7], an analytical approach to model the Block ACK in IEEE 802.11e was proposed without accounting for the hidden terminal problem. The multi-rate feature in the same environment was also not considered. Further, the packet loss due to errors in the channel was not taken in consideration. Finally, the use of EDCA procedures, like several virtual queues, for the several classes of service are also not considered, i.e., this work did not consider the IEEE 802.11e standard at all.

From the simulation approaches presented in the literature the one that is most similar to the approach we propose here is the one presented in [8]. In [8], several combinations for the block size are presented, where a scheduler based on the delay and amount of data in the buffer is proposed. Out proposal is an improvement of this approach and provides a more extensive study on the block size whilst considering use-perceived QoS.

The remaining of the paper is organized as follows. In Section II, a brief introduction to the IEEE 802.11e standard is presented along with the rationale behind the Block ACK procedure. Section III presents the state transition diagram of the simulator, and the list of events. In Section IV, details are given on the IEEE 802.11a standard for the physical layer, the one considered in this work. Section V discusses the

Manuscript received December 15, 2008. The authors would like to acknowledge the fruitful contributions given by António Grilo for the implementation of the PHY part of the simulator. This work was partially funded by CROSSNET (a Portuguese Foundation for Science and Technology POSC project with FEDER funding), by IST-UNITE, by the FCT PhD grant SFRH/BD/28517/2006, by Fundação Calouste Gulbenkian, and by “Projecto de Re-equipamento Científico" REEQ/1201/EEI/ 2005.

Orlando Cabral, Alberto Segarra, and Fernando Velez are with the Instituto de Telecomunicações, Universidade da Beira Interior, Covilhã, Portugal (phone: +351 275329953; fax: +351 275329972; e-mail: orlandoc@ubi.pt, vyniard@gmail.com, fjv@ubi.pt).

(Advance online publication: 17 February 2009)

hypothesis for system and scenarios, including details on traffic parameters. Section VI presents simulation results obtained for several scenarios in the absence and presence of the Block ACK procedure. Finally, conclusions are presented in Section VII as well as suggestion for future work.

II. IEEE 802.11E

A. User priorities and Access Categories

IEEE 802.11e provides medium access control (MAC) enhancements to support local area network (LAN) applications with QoS requirements in wireless environments. By using eight different user priorities (UPs), the so-called enhanced distributed channel access (EDCA) provides differentiated, distributed access to the medium for stations by using four different access categories (ACs) that provide support for the delivery of traffic with various UPs at the stations. The AC is derived from the UPs as presented in Table I. This differentiation is achieved by varying the following different values for the UP: i) the amount of time a STA senses the channel to be idle before backoff or transmission, ii) the length of the contention window to be used for the backoff, or iii) the duration a STA may transmit after it acquires the channel.

Details on the CSMA/CA protocol, transmission opportunities (TXOP), and inter-frame spaces (IFS) are presented in [1, 2]. The backoff time and the backoff procedure are addressed in [1, 3], as well as the description of the details on the network allocation vector (NAV), and the use of RTS/CTS with fragmentation.

TABLE I. MAPPING BETWEEN USER PRIORITIES AND ACS.

	UP (Same as 802.1D user priority)	802.1D Designation	AC	Designation
Lowest	1	BK	BK	Background
	2	—	BK	Background
	0	BE	BE	Best Effort
	3	EE	BE	Best Effort
	4	CL	VI	Video
	5	VI	VI	Video
	6	VO	VO	Voice
Highest	7	NC	VO	Voice

B. Block Acknowledgment

The Block Acknowledgment (Block ACK) mechanism improves channel efficiency by aggregating several acknowledgments into one frame. There are two types of Block ACK mechanisms: immediate and delayed. Immediate Block ACK is suitable for high-bandwidth, low-latency traffic while the delayed Block ACK is suitable for applications that tolerate moderate latency. The QoS Station (QSTA) with data to send using the Block ACK mechanism is referred to as the originator while the receiver of that data is the recipient.

The Block ACK mechanism is initialized by an exchange of Add Block Acknowledgment (ADDBA) Request/ Response frames. After initialization, blocks of QoS data frames can be transmitted from the originator to the recipient. A block may be started within a polled TXOP or by winning EDCA contention. The number of frames in the block is limited, and the amount of state that is to be kept by the recipient is bounded. The MPDUs within the block of frames are acknowledged by a BlockACK control frame, which is requested by a Block ACKReq control frame.

Figure 1 illustrates the message sequence chart for the setup, data and Block ACK transfer, and the teardown of the Block ACK mechanism.

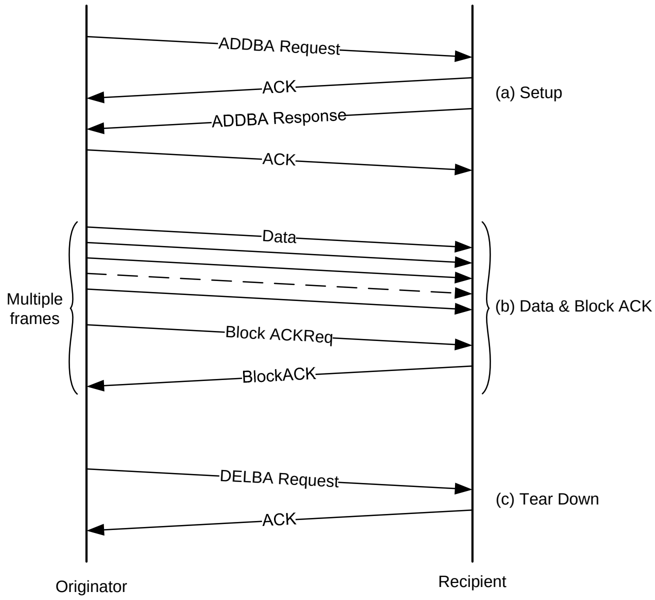

Figure 1. Block ACK sequence

C. Setup and modification of the Block ACK parameters

A QSTA that intends to use the Block ACK mechanism for the transmission of QoS data frames to a peer should first check whether the intended peer QSTA is capable of participating in Block ACK mechanism by discovering and examining its buffer of Block ACK. If the intended peer QSTA is capable of participating, the originator sends an ADDBA Request frame indicating the traffic identifier (TID) for which the Block ACK is being set up. The Block ACK Policy and Buffer Size fields in the ADDBA Request frame are advisory and may be changed by the recipient. The receiving QSTA shall respond by an ADDBA Response frame. The receiving QSTA, which is the intended peer, has the option of accepting or rejecting the request. When the QSTA accepts the request, then a Block ACK agreement exists between the originator and recipient. In this case, it indicates the type of Block ACK and the number of buffers that it shall allocate for the support of this block. If the receiving QSTA rejects the request, then the originator shall not use the Block ACK mechanism.

Once the Block ACK exchange has been set up, data and ACK frames are transferred by using the procedure presented in Figure 1.

D. Data and acknowledgment transfer

After setting up for the block exchange following the previous procedure, the originator may transmit a block of QoS data frames separated by SIFS periods, with the total number of frame not exceeding the Buffer Size subfield value in the associated ADDBA Response frame. Each of the frames shall have the ACK Policy subfield in the QoS Control field set to Block ACK. The RA field of the frames shall be the recipient's unicast address. The originator requests acknowledgment of outstanding QoS data frames by sending a Block ACKReq frame. The recipient shall maintain a Block ACK record for the block.

Subject to any constraints in this sub clause about permitted use of TXOP according to the channel access mechanism used, the originator may:

- Separate the Block ACK and Block ACKReq frames into separate TXOPs;
- Split a Block frame across multiple TXOPs;
- Sequence or interleave MPDUs for different RAs within a TXOP.

A protective mechanism (such as transmitting using RTS/CTS) should be used to reduce the probability of other STAs to transmit during the TXOP. If no protective mechanism is used then the first frame that is sent as a block shall have a response frame and shall have the Duration field set so that the NAVs are set to appropriate values at all STAs in the QBSS.

The originator shall use the Block ACK starting sequence control to signal the first MPDU in the block for which an acknowledgment is expected [1]. MPDUs in the recipient's buffer with a sequence control value that precedes the starting sequence control value are called preceding MPDUs. The recipient shall reassemble any complete MSDUs from buffered preceding MPDUs, and indicate these to its higher layer. The recipient shall then release any buffers held by preceding MPDUs. The range of the outstanding MPDUs (i.e., the reorder buffer) shall begin on an MSDU boundary. The total number of frames that can be sent depends on the total number of MPDUs in all the outstanding MSDUs. The total number of MPDUs in these MSDUs may not exceed the reorder buffer size in the receiver.

The recipient shall maintain a Block ACK record consisting of originator address, TID, and a record of reordering buffer size indexed by the received MPDU sequence control value. This record holds the acknowledgment state of the data frames received from the originator.

If the immediate Block ACK policy is used the recipient shall respond to a Block ACKReq frame with a Block Ack frame. If the recipient sends the Block ACK frame, the originator updates its own record and retries any frames that are not acknowledged in the Block ACK frame, either in another block or individually. If the delayed Block ACK policy is used, the recipient shall respond to a Block ACKReq frame with an ACK frame. The recipient shall then send its Block ACK response in a subsequently obtained TXOP. Once the contents of the Block ACK frame have been prepared, the recipient shall send this frame in the earliest possible TXOP using the highest priority AC. The originator shall respond with an ACK frame upon receipt of the Block ACK frame.

The Block ACK frame contains acknowledgments for the previous MPDUs. In the Block ACK frame, the QSTA acknowledges only the MPDUs starting from the starting sequence MPDU until the last MPDU that has been received, and the QSTA shall set bits in the Block ACK bitmap corresponding to all other MPDUs to 0. If the Block ACK frame indicates that an MPDU was not received correctly the originator shall retry that MPDU subject to that MPDUs appropriate lifetime limit.

A typical Block ACK frame exchange sequence using the immediate Block ACK is presented in Figure 2 while a typical Block ACK sequence using the delayed Block ACK is presented in Figure 3. The subsequent Block ACK request starting sequence number shall be higher than or equal to the starting sequence number (modulo 212) of the immediately preceding Block ACKReq frame.

The originator may continue to transmit MPDUs to the recipient after transmitting the Block ACKReq frame, but only before receiving the Block ACK frame (applicable only to delayed Block ACK). The bitmap in the Block ACK frame shall include the status of frames received between the start sequence number and the transmission of the Block ACKReq frame. A recipient sending a delayed Block ACK frame may update the bitmap with information on QoS data frames received between the receipt of the Block ACKReq frame and the transmission of the Block ACK frame.

If there is no response to the Block ACKReq frame (i.e., neither a Block ACK nor an ACK frame), the originator may retransmit the BlockACKReq frame within the current TXOP (if time permits) or within a subsequent TXOP. MSDUs that are sent using the Block ACK mechanism are not subject to retry limits but only to MSDU lifetime. The originator needs not to set the retry bit for any possible retransmissions of the MPDUs. The Block ACKReq frame shall be discarded if all MSDUs referenced by this Block ACKReq frame have been discarded from the transmit buffer due to the expiration of their lifetime limit.

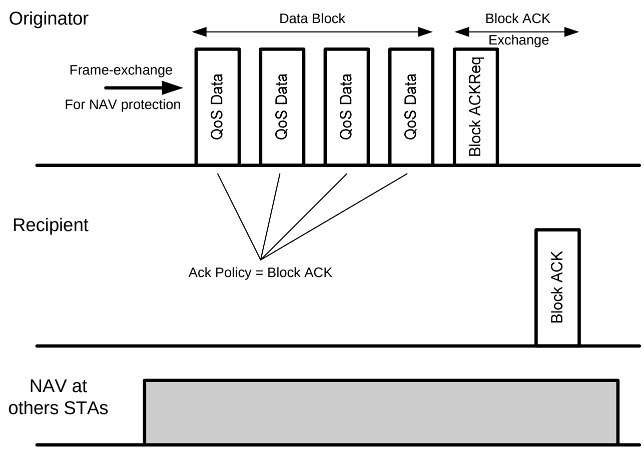

Figure 2. Inmediate Block ACK.

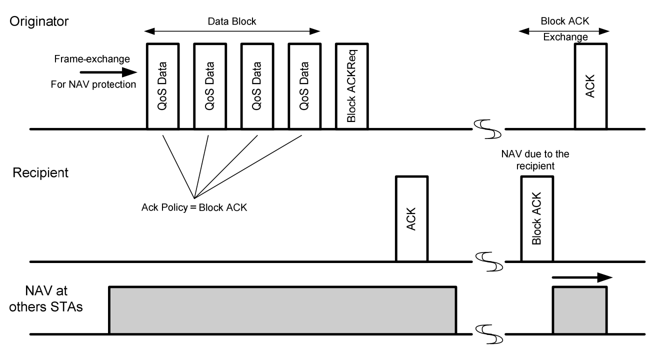

Figure 3. Delayed Block ACK.

In order to improve efficiency, originators using the Block ACK facility may send MPDU frames with the ACK Policy subfield in QoS control frames set to Normal ACK if only a few MPDUs are available for transmission. The Block ACK record shall be updated irrespective of the ACK Policy subfield in the QoS data frame for the TID with an active Block ACK. When there is sufficient number of MPDUs, the originator may switch back to the use of Block ACK. The reception of QoS data frames using Normal ACK policy shall not be used by the recipient to reset the timer to detect Block ACK timeout. This allows the recipient to delete the Block ACK if the originator does not switch back to using Block ACK.

E. Receive buffer operation

Upon the receipt of a QoS data frame from the originator for which the Block ACK agreement exists, the recipient shall buffer the MSDU regardless of the value of the ACK Policy subfield within the QoS Control field of the QoS data frame.

The recipient shall always indicate the reception of MSDU to its MAC client in order of increasing sequence number.

F. Teardown of the Block ACK mechanism

When the originator has no data to send and the final Block ACK exchange has completed, it shall signal the end of its use of the Block ACK mechanism by sending the DELBA frame to its recipient. There is no management response frame from the recipient. The recipient of the DELBA frame shall release all resources allocated for the Block ACK transfer.

The Block ACK agreement may be turn down if there is no Block ACK, Block ACKReq, or QoS data frames (sent under Block ACK policy) for the Block ACK's TID received from the peer until the Block ACK timeout instant.

III. STATE TRANSITION DIAGRAM

The state transition diagram used to build the simulator is presented in Figure 4. The following events cause transition/ change of the machine state:

NEW_PCK_BK a new packet of BK is generated;
NEW_PCK_BE a new packet of BE is generated;
NEW_PCK_VI a new packet of VI is generated;
NEW_PCK_VO a new packet of VO is generated;
STOP_LTN_DIFS end of the AIFS period for sensing the medium;
STOP_LTN_SIFS end of the SIFS period for sensing the medium;
TIME_SLOT the STA decrements the *backoff_value*;
START_TX the station starts to transmit;
STOP_TX end of the transmission;
START_RX begin of the reception;
STOP_RX end of the reception;
ACK_OK the ACK was received;
ACK_NOK the ACK was not received.

A detailed description of the possible states for the "machine" nodes, the simulation entities, the simulation variables, and the functions for events is presented in [9].

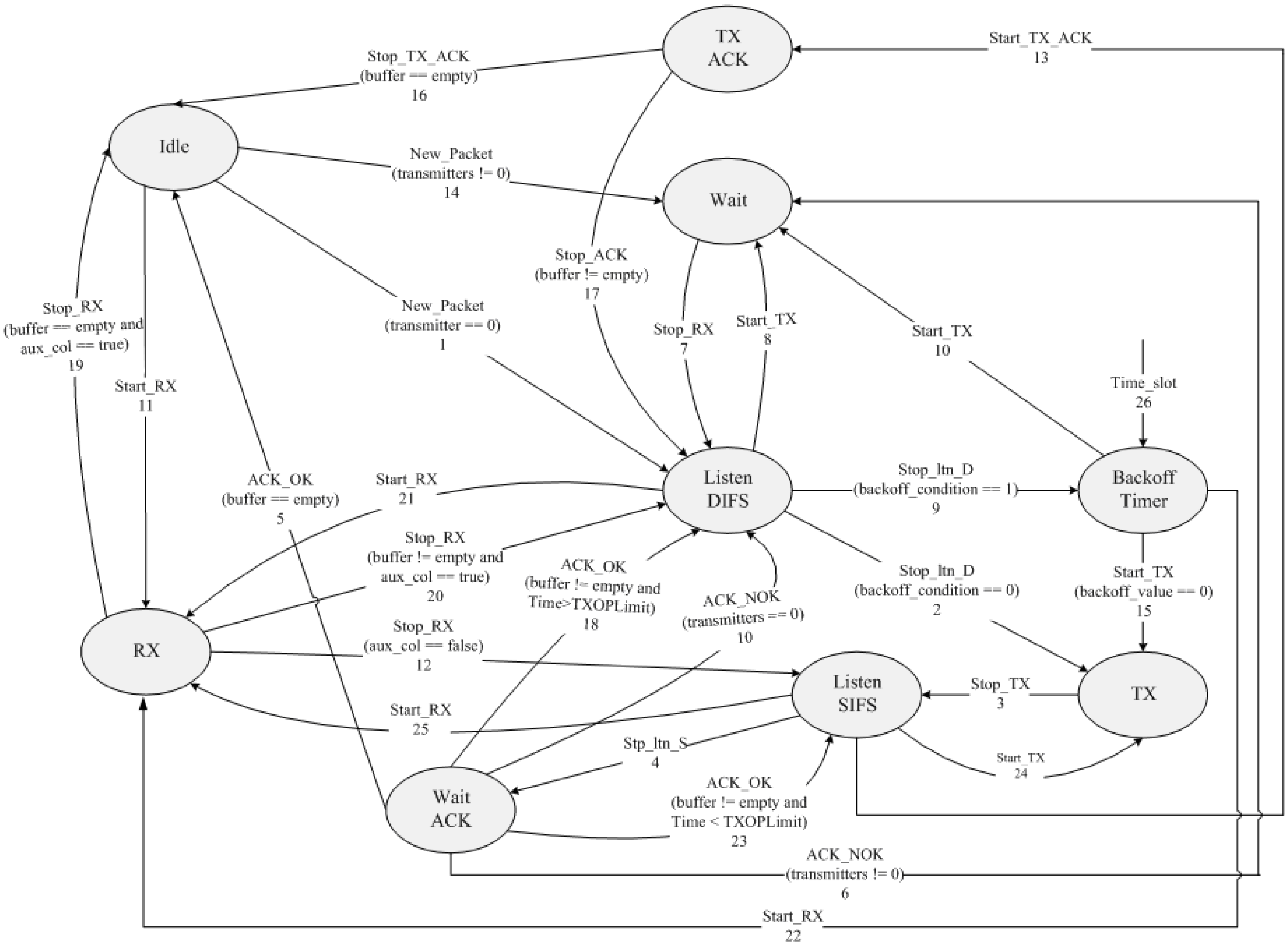

Figure 4. State transition diagram (the incoming arrow for the Backoff_Timer state means a transition from and to the same state).

IV. PHYSICAL LAYER

In this work we consider the IEEE 802.11a physical layer, PHY, specification [10, 11]. It defines an Orthogonal Frequency Division Multiplexing (OFDM) based PHY that operates in the 5 GHz frequency bands, and is able to achieve bit-rates up to 54 Mbps.

IEEE 802.11a specifies 8 different transmission modes, Table II obtained with different combinations of modulation and convolutional code rate. Each transmission mode corresponds to a different bit-rate. In Table II, the respective number of byte transmitted in one OFDM symbol (Bytes-per-Symbol, BpS). The convolutional encoder always encodes data with code rate 1/2. The 3/4 and 2/3 codes are derived from the original 1/2 code by a technique called puncturing. Puncturing is a procedure for omitting some of the encoded bits in the transmitter, and inserting a dummy “zero” metric into the convolutional decoder at the receiver, in place of the omitted bits. This technique is a simpler and more efficient way of generating a higher code rate.

In our simulations, we estimate the received power and the value of the signal-to-interference-plus-noise ratio (*SINR*) experienced at the receiver, based on the last reception of that machine, a procedure very similar with the one from [12]. In this algorithm, the sender chooses the bit-rate that achieves the highest goodput taking into account the *SINR* estimate. More details on the implementation of PHY into the simulator can be found in [13]. Table II also presents the percentage of usage for each transmission mode and the corresponding radius within the transmission modes are made available. The corresponding Bird’s-eye view is presented in Figure 5.

TABLE II. IEEE 802.11A PHY MODES.

Mode	Modula-tion	Code Rate	Bit rate	BpS	%	Cell radius [m]
1	BPSK	1/2	6 Mbps	3	0.4	The
2	BPSK	3/4	9 Mbps	4.5		
3	QPSK	1/2	12 Mbps	6	8.4	33.82
4	QPSK	3/4	18 Mbps	9	26.0	28.30
5	16-QAM	1/2	24 Mbps	12	19.2	22.78
6	16-QAM	3/4	36 Mbps	18	20.1	19.13
7	64-QAM	2/3	48 Mbps	24	3.8	14.35
8	64-QAM	3/4	54 Mbps	27	22.0	13.24

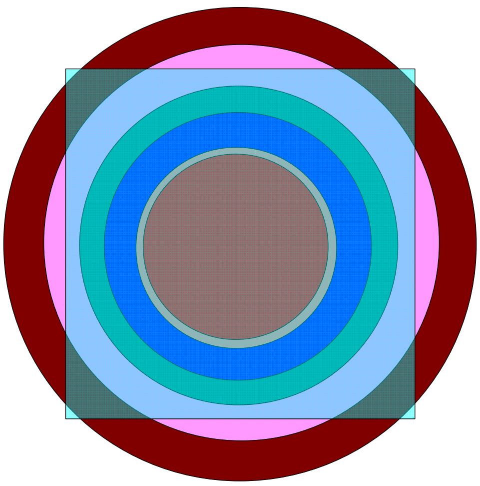

Figure 5. Bird’s view of cell area.

V. SYSTEM, SCENARIO AND ASSUMPTIONS

Lets consider a cellular WiFi system where each cell has a set $N+1$ IEEE 802.11e stations communicating through the same wireless channel. While station 0 is the Access Point or QoS Access Points (QAP), the other N wireless terminals are QSTAs. The propagation time is assumed to be absorbed by some mechanisms of the IEEE 802.11, like Simple Inter-frame Spacing (SIFS). Each station has four buffers whose size depends on the kind of service being dealt in order to guarantee a given value for the goodput (payload of the packet), and there is a correspondence between the buffer and the ACs.

This buffer will be filled with a MSDUs generated that characterises the service being dealt in it. If the MSDU is larger than a fragmentation threshold, it will be fragmented. In order to cope with service quality the packet transmission follows the Enhanced Distributed Channel Access (EDCA) IEEE 802.11e MAC procedure. Due to collisions or interference a packet may not be correctly received. The number of collision is represented by a global variable that checks whether there is more than one user transmitting simultaneously. The interference issues are addressed by using a radio propagation model. Each packet exits the buffer only after the reception of an acknowledgement, or if it has suffered more than a collision threshold.

The users are assumed to be static, and are distributed uniformly in a square area of 2500 square meter. The topology to be implemented consists of several wireless stations and an Access Point (AP). Three types of traffic sources were chosen, namely high priority voice (VO), medium priority video (VI) and low priority FTP data, as background traffic (BK). The traffic sources parameters are presented in Table III, as well as the access categories (ACs) of each type of traffic.

TABLE III. TRAFFIC PARAMETERS [10].

AC	Voice (VO)	Video (VI)	Background (BK)
Packet size	1280 bit	10240 bit	18430 bit
Packet interval	20 ms	10 ms	12.5 ms
Usage	50 %	30 %	20 %
symmetry	symmetric	asymmetric (downlink)	asymmetric (downlink)

We implemented the Block ACK procedure with minor modification to the existing features like TXOP and RTS/CTS as already explained and without disturbing the overall TXOP and RTS/CTS procedure. When a TXOP is gained, the transmission starts and the origin will know if a Block ACK procedure is implemented with this destination. If this is true, it will not wait for an acknowledgement, but just a SIFS and start the next transmission for the same destination or not depending on which is the next packet in the buffer. Figure 6 presents this procedure for destination 1, which has the Block ACK procedure implemented. Communications to destination 2 does not support Block ACK. At the beginning of a transmission for a user with active Block ACK procedure, if the block size threshold is reached, a block ACK request packet is added to the buffer to the top-1 place in the queue, to be transmitted as the next in line packet.

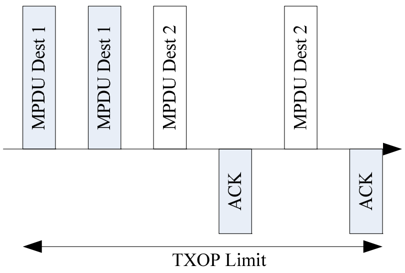

Figure 6. Illustration of the Block ACK procedure within a TXOP.

VI. Initial Simulation Results

A. Standalone services

The objective of the simulations was to investigate the gain of using the Block ACK procedure both for a single service and for a mixture of services. In terms of grade of service, the voice application supports delays up to 30 ms, the video application supports delays up to 300 ms, while the delay for background applications can be up to 500 ms [14].

Our simulations consider one access point with several client machines. Results include packet delay, in milisecond, goodput, in bits per second, and channel utilization in percentage. The results consider the simultaneous requirements from MAC and PHY layers. For each parameter, they are obtained as the average of 20 simulations (each with different random seeds). Simulation results with MAC layer alone were presented in [9].

Packet delay is the period of time between the instant at which the packet arrives to the buffer and the instant at which the packet is successfully transmitted. Figure 7 presents results for delay for stand alone voice (VO), background (BK) and video (VI). Hence, in the different experiences, our QAP supports up to 40 voice users, 18 video users, and 11 background users, with an appropriate degree of QoS.

Another performance measure is the maximum achieved goodput achievable for a given channel capacity. It is certain that a fraction of the channel capacity is used up in form of overhead, acknowledgments, retransmission, token delay, etc. Channel capacity is the maximum possible data rate, that is, the signalling rate on the physical channel. It is also known as the data rate or transmission rate, assumed to be variable, between 6 and 54 Mbit/s. Goodput is the amount of “user data” that is carried by the wireless network. The results for goodput as function of the number of station are presented in Figure 8 for the same set of experiences. Its maximum value is 16 Mbit/s.

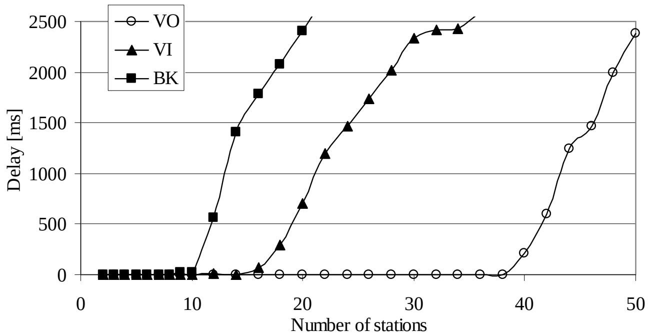

Figure 7. Delay for each application as a function of the number of station.

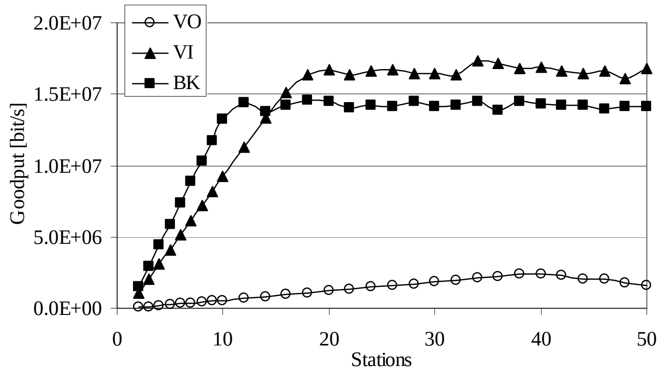

Figure 8. Goodput as a function of the number of node for VO, VI, and BK applications.

Due to the scarcity of wireless bandwidth, it is important to evaluate its relative usage, i.e., its average utilization. For this purpose, we have computed the average data rate for the client stations. Since the distribution of the users in the square is uniform, it is easy to compute the probability of a given transmission mode, and the average available data rate for the client stations, according to the Bird’s eye view of the cell from Figure 5. The channel utilization is the ratio of goodput over the average data rate, and is presented in Figure 9 (as a function of the number of station).

The highest value obtained for the utilization, the ratio between the goodput and the average data rate, is around 80% for VI, while the lowest one occurs for VO traffic. This is justified because the packet size is higher for VI than for VO.

As previously mentioned, the Block ACK mechanism improves the channel efficiency by aggregating several acknowledgments into one frame. To investigate the proposed Block ACK procedure for the stand-alone service a Block ACK procedure with a block size of 16 fragments was adopted. This procedure was simulated only for BK and VI traffic. For VO application it is certain that Block ACK is not a solution because of the large delays that occur when the buffer is filled with 16 packets, i.e., the delay of the first one would be 320 ms.

Figure 10 presents a comparison of the delay between the cases of presence and absence of Block ACK for BK and VI traffic. It starts to increase around 12 stations for BK and at 16 stations for VI and increases more for a higher number of stations. As expected, the delay is lower when the Block ACK procedure is used. The improvement (reduction) for BK traffic is 300 ms, on the average, after 12 stations, while for VI traffic it is 420 ms, on the average, after 16 stations.

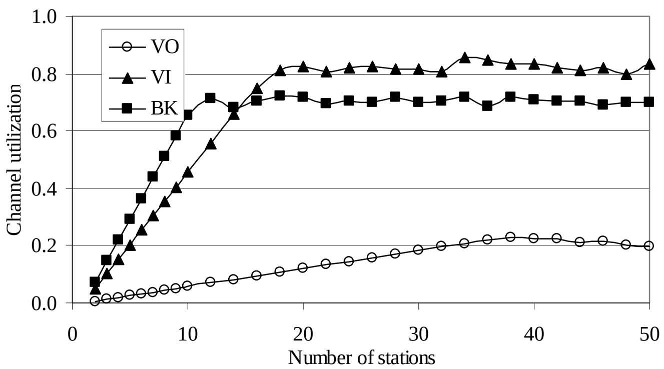

Figure 9. Channel utilization, as a function of the number of node for VO, VI, and BK applications.

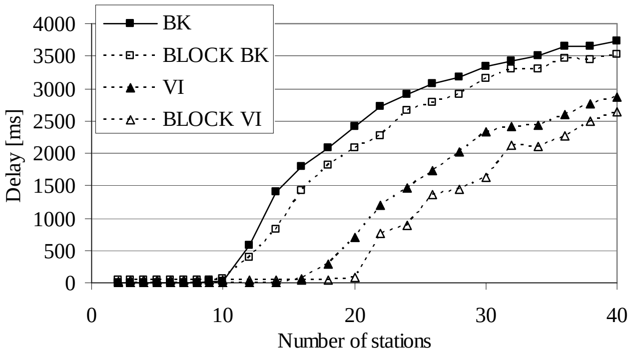

Figure 10. Delay for BK and VI with and without Block ACK.

The variation of the goodput with the number of station is presented Figure 11 for BK and VI for simulations lasting 10 s. It is higher with Block ACK procedure than without it. The improvement is of 2 Mb/s and 2.2 Mb/s in average, after 12 stations, for BK traffic and after 16 stations for VI, respectively.

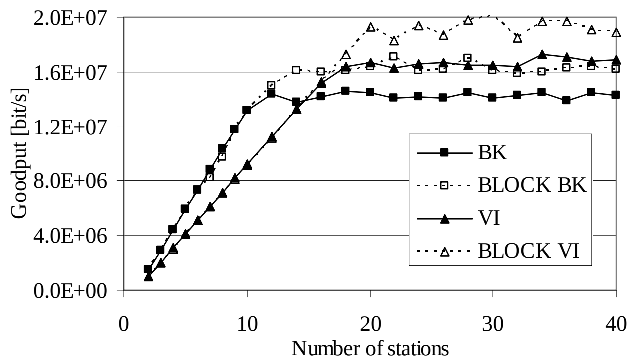

Figure 11. Goodput for BK and VI with and without Block ACK.

B. *Mixtures of applications*

This part of the work intends to explore which should be the policy regarding the block size with mixtures of applications. For the purpose we have investigated the Block ACK policy for a video service, an application that is delay-sensitive. Additionally, we have investigated the Block ACK block size for background service that is not delay sensitive. Simulations were performed for 100 s and around 40 times for each number of stations, {15, 20, 25, 30, 35, 40, 45}, and each block size, {4, 8, 12, 16}. The users started the negotiation to initiate Block ACK procedure in the beginning of the simulation so all the packets of video and background traffic were transmitted within the Block ACK procedure.

The use of the Block ACK procedure in a scenario with a mixture of simultaneous applications was further studied. On the one hand, the overhead caused by the negotiation to establish the Block ACK procedure, and to maintain the Block ACK causes bad performance when a small block size is used. On the other, the packet losses caused by the voice users with higher priority, influences the overall QoS on the system and mainly on video. The reason is the following: if a packet is not correctly received at a given time instant the next packets sent within this block will have the delay of the badly sent packet (after successfully transmitted) added to their delay, Figure 12.

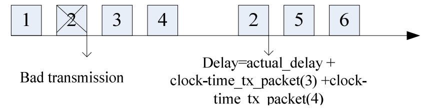

Figure 12. Procedure to count delay.

The packet retransmissions cause therefore an increase on the delay. Figure 13 presents the variation of the delay with the number of station for each access category. For BK traffic the delay starts to increase considerably with more than 20 stations (12 in the standard procedure). This service class will rarely transmit since the voice and video traffic will always be scheduled first. For video traffic the delay starts to increase for more than 35 stations. In contrast with BK and VI, VO applications present a negligible delay.

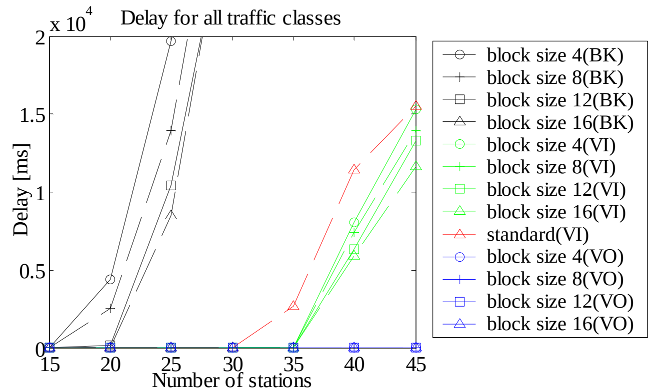

Figure 13. Delay for all services with Block ACK implemented for VI and BK traffic.

The application that mostly suffers from the degradation on delay is therefore the video one. Figure 14 presents results for the variation of the video application delay with the number of station for various block sizes. Regardless the number of station, the block ACK procedure reduces the delay. Certain sensitivity is observed for a block size of 16, which is the size that gives the highest delay for a total of 15-25 stations. Although a block size of 12 gives better results 35 stations, for a larger number of station the best block size is 16. Smaller block sizes are not that efficient in decreasing the delays anymore. Results for 40 and 45 stations advise the use of block size 16, although the network is already overloaded (and the delay will just keep growing up to infinity). Figure 15 illustrates this fact.

When 30 stations are being served in the system the results for the delay with Block ACK is near 5 ms for all block sizes, while without Block ACK the delay goes up to 80 ms.

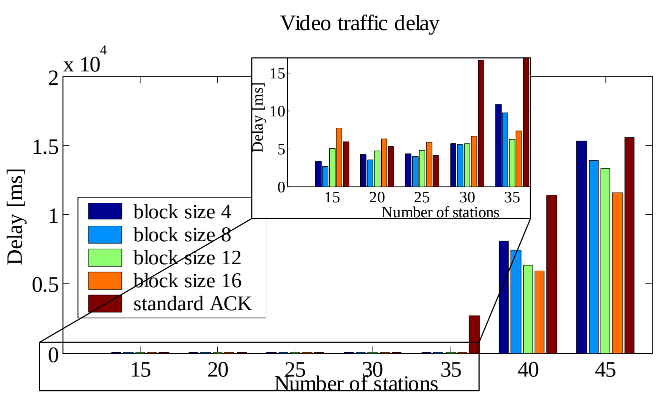

Figure 14. Delay for video traffic.

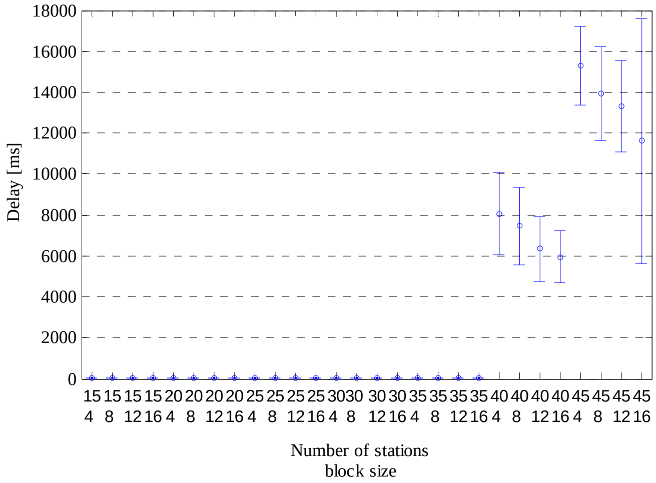

Figure 15. Confidence intervals for the delay of video traffic.

It is also worthwhile to note that for 35 stations the difference is even higher. A minimum near 5 ms is obtained for the delay for a block size of 12 while without Block ACK the delay is near 2.3 s. When less then 40 stations are present the confidence intervals are very narrow for all sizes of the buffers. When there are 40 or more stations the amplitude of the confidence intervals increase. One can therefore extrapolate that there are stations that manage to transmit the packet and have a small delay while others transmit from time to time, causing very high delays. The behaviour observed in Figure 14 occurs due to the overhead caused by the Block ACK request, and the delays caused by bad receptions affected mostly the block size 16, but providing lower values for the delay for higher loaded scenarios. The solution is neither to use a large block size (as large as 16) nor a small block size (as low as 4). The former increases the delay causing problems to the application using such packets, while the latter causes some unnecessary overhead by requesting very often block ACK requests.

Figures 16 and 17 present the collisions and packet loss rate. It can be extracted from this results that the number of collisions per packet and the packet loss rate do not significantly depend of the block size. The advised block size shall be 12 since it is the one that provides lower delay in a scenario where the load is still manageable, at least for the case where Block ACK is used. Figure 18 presents the results of the goodput in the system for the downlink. Without the Block ACK the maximum goodput achieved is near 11 Mbit/s, while with Block ACK is near 14 Mbit/s for block sizes of 12 and 16.

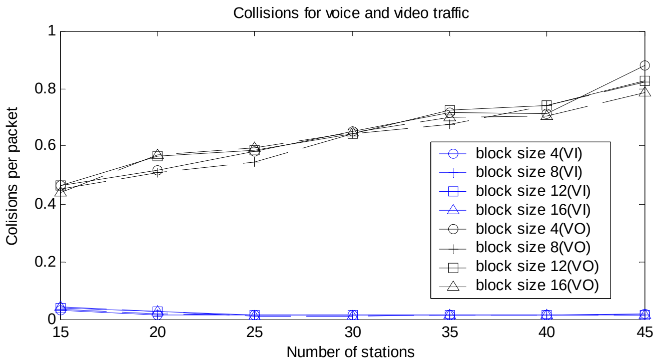

Figure 16. Collisions for video and voice traffic.

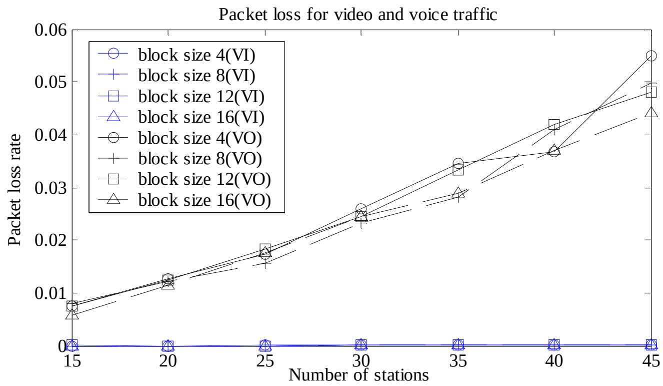

Figure 17. Packet loss for voice and video traffic.

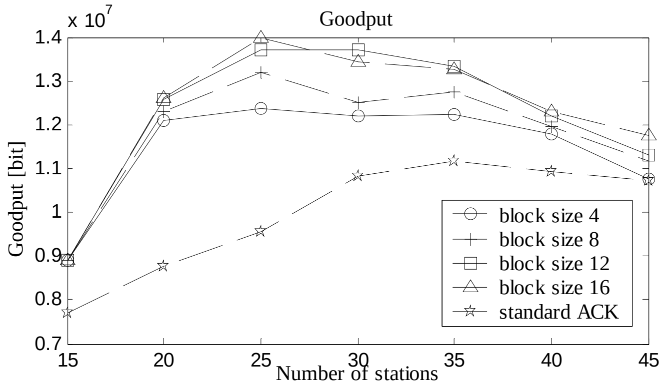

Figure 18. Total goodput with Block ACK implemented for video and background traffic.

The decreasing behaviour after 25 stations occurs due to the high number of collision, and as the background traffic has low priority ends up not being transmitted giving its turn to voice and video users. As voice traffic provides higher overhead the resulting goodput is lower.

Figure 19 presents the goodput only for the video service class. Only after 30 stations the achieved throughput is different when using and not using Block ACK.

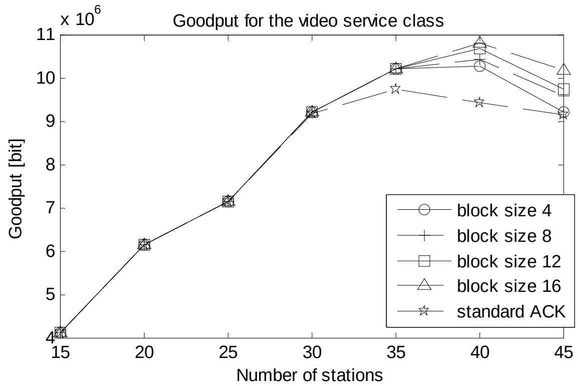

Figure 19. Video goodput with Block ACK implemented for video traffic.

The highest goodput is found for block size 16, and is more than 10% higher relatively to the case of use of standard ACK. Note that the increasing behaviour of the goodput is verified up to 40 stations.

VII. Conclusions and Future Work

The influence of the Block ACK size as a policy to decrease delays and ensure a stable wireless system was addressed. The investigation was based on tuning several system parameters and comparing the effect of the presence and absence of Block ACK. In the presence of block ACK several policies were explored.

In a single service scenario although for lower values of the number of station, the use of Block ACK leads to a slightly worst system performance, the Block ACK procedure provides an improvement in highly loaded scenarios. The improvement is of 2 Mb/s and 2.2 Mb/s in average, for BK traffic and VI traffic, respectively.

In a scenario with mixture of services, the most advised block size is 12 since it is the one that provides lower delays in a highly loaded scenario while the users are still within the capacity of the AP. In comparison with the absence of Block ACK the number of supported user increases from 30 to 35. Note that 35 stations is the total number of VO plus VI and BK user. In the multi-service case, higher goodputs are also obtained for VI traffic. This is mainly due to the high payload of the frames, as CW[VI]=2·CW[VO]. When the number of station is small, the goodput for VO applications is the lowest. However, for a higher number of stations the lowest goodput occurs for BK traffic. For VO applications the goodput is always increasing while for BK and VI traffic there is a minimum value for the number of station after which the individual goodput of each station starts to decrease, explained by the small values of CW and AIFS for VO.

Future work will include testing policies that provide access to the medium while ensuring improved grade of service based on the channel *SINR*, values of the delay, and throughput.

REFERENCES

[1] IEEE, IEEE Std. 802.11e; Wireless LAN Media Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE, New York, NY, USA, Nov. 2005.

[2] IEEE, IEEE Std. 802.11; Wireless LAN Media Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE, New York, NY, USA, 1999.

[3] Anand R. Prasad, Neeli R. Prasad, *802.11 WLANs and IP Networking, security, QoS, and mobility*, Artech House, London, UK,2005.

[4] Jijun Luo Mukerjee, R. Dillinger, M. Mohyeldin, E. Schulz, E., "Investigation of radio resource scheduling in WLANs coupled with 3G cellular network," *IEEE Communications Magazine*, Vol. 41, No. 6, pp. 108-115, June 2003.

[5] Tianji Li, Qiang Ni, Thierry Turletti, and Yang Xiao, "Performance Analysis of the IEEE 802.11e Block ACK Scheme in a Noisy Channel," in *Proc. of IEEE BroadNets 2005 - The Second International Conference on Broadband Networks*, Boston, MA, USA, Oct. 2005

[6] Tianji Li, Qiang Ni, and Yang Xiao, "Investigation of the Block ACK Scheme in Wireless Ad-hoc Networks," Wiley Journal of Wireless Communications and Mobile Computing, Vol. 6, No. 6, pp. 877-888, Aug. 2006.

[7] Ilenia Tinnirello and Sunghyun Choi, "Efficiency Analysis of Burst Transmissions with Block ACK in Contention-Based 802.11e WLANs," in *Proc. of ICC 2005 - IEEE International Conference on Communications*, Seoul, Korea, May 2005.

[8] V. Scarpa, G. Convertino, S. Oliva, C. Parata, "Advanced Scheduling and Link Adaptation Techniques for Block Acknowledgement," in *Proc. of 7th IFIP International Conference on Mobile and Wireless Communications Networks (MWCN 2005)*, Marrakech, Morocco, Sep. 2005.

[9] Orlando Cabral, Alberto Segarra, Fernando J. Velez, "Event-Driven Simulation for IEEE 802.11e Optimization," *IAENG International Journal of Computer Science (IJCS)*, Vol. 35 , No. 1, pp. 161 -173 , Feb. 2008.

[10] Quiang Ni, "Performance Analysis and enhancements for IEEE 802.11e Wireless Networks," *IEEE Network*, Vol. 19, No. 4, July/Aug. 2005, pp. 21-27.

[11] IEEE, IEEE Std. 802.11a; Wireless LAN Media Access Control (MAC) and Physical Layer (PHY) Specifications: High-speed Physical Layer in the 5 GHz Band, IEEE, New York, NY, USA, Sep. 1999.

[12] D. Qijao, S. Choi, "Goodput enhancement of IEEE 802.11a wireless LAN via link adaptation," in *Proc. of IEEE ICC'2001 - International Conference on Communications*, Helsinki, Finland, June 2001.

[13] A Grilo, *Quality of Service in IP-based WLANs*, Doctoral thesis, Instituto Superior Técnico, Technical University of Lisbon, Lisbon, Portugal, June 2004.

[14] N. Anastácio, F. Merca, O. Cabral, F. J. Velez, "QoS Metrics for Cross-Layer Design and Network Planning for B3G Systems," in *Proc. of IEEE ISWCS2006 - Third International Symposium on Wireless Communication Systems*, Valencia, Spain, Sep. 2006.