A Parameter Injection Algorithm for Real-time Traffic in 802.11 Open Access Networks

Muslim Elkotob and Sahin Albayrak

DAI-Labor, TU-Berlin, Germany {<firstname>.<lastname>@dai-labor.de}

Abstract— An Open Access Network (OAN) is characterized by the fact that private WLANs and broadband access lines are made available for public use, enabling bypassing mobile users to profit from continuous coverage. Our insight is that in wireless OANs, the overhead required for the launching or reconfiguration of a real-time session can be drastically reduced. We propose an algorithm which performs parameter injection into a multimedia data stream instead of following the classical negotiation path typically used. A solution to change the session characteristics of an application that adapts to varying network conditions was developed, including a QoS solution with SIP and MIP to optimize the session characteristics by taking into account the changed conditions in the network. With our approach, we achieve a reduction of a factor of 2.5 for session signaling delay, and excellent jitter values in the range of 10-15 milliseconds as our implementation results show.

I. INTRODUCTION AND MOTIVATION

This paper focuses on the design, implementation, and evaluation of a new algorithm for managing signaling for automatically updating real-time session settings in a multi-cell open access network. In such a network a station traverses open WLAN access points which are adjacent and provide continuous coverage.

The current trend in telecommunications is constantly witnessing an increase in the demand for resources by mobile applications and services. Various mechanisms have been proposed to cope with this scarcity of resources in the wireless world [9], [10], [11], [12], [13]. Real-time applications are not only resource consuming, but also have very rigid restrictions on network quality parameters such as jitter and delay [24]. This makes it especially difficult to handle this class in OANs. Due to its abundance and ease of deployment for operators, as well as its portability, IEEE 802.11 [14] is an ideal choice to build an interconnected mesh-like network which shall provide coverage and connectivity to mobile users.

During mobility, our capacity distribution algorithm (CDA), may decide to redistribute the capacity over all associated WLAN stations. Various reasons may trigger this redistribution, including the situation that certain stations jeopardize the profile conditions committed to others. A redistribution of capacity affects the network capacity that can be used by a mobile station and this in turn may affect the quality experienced by the end-user. Actions may be needed to minimize the effects, realized by changing the settings of real-time applications. The algorithm that optimizes QoS settings of real time sessions dynamically copes with capacity redistribution effects.

The algorithm takes into account the SDP/RTP [6] configuration algorithm for handovers and uses mapping tables to compute whether changed committed profiles require updates for running applications. It then performs these updates if needed. The updates may be realized by selecting a different codec and may require communication with the server to inform it about the changes.

The paper is structured as follows. After this introductory section, Section II summarizes related work. Section III explains our approach for QoS brokerage and capacity management in WLAN-based Open Access Networks. Section IV then moves on to introduce the parameter injection algorithm in itself in addition to wrapping up the resource brokerage process for real-time traffic. Section V evaluates the results achieved in this paper quantitatively, whereby the designed algorithm is implemented and tested for performance. Section VI then concludes the paper.

II. RELATED WORK

Zhang et al. propose a path-oriented, quota-based QoS brokerage method which aims at increasing the overall call processing capability of the bandwidth broker box [4]. They rely on path-level admission control and link-level bandwidth allocation. They differ from us in the sense that their architecture is very centralized, having one box with a single bandwidth broker in the deep backend; whereas our approach uses many last-mile QoS brokers which are light-weight and optimize capacity based on air-time quota [26].

Duan et al. propose for the QoS bandwidth brokerage issue to decouple the QoS control plane from the packet forwarding plane [3]. They store QoS reservation states and manage all those states inside the bandwidth broker in addition to performing aggregate and per-flow management. Again their approach lacks the light-weight property and is too centralized.

Krishnamurthy et al. focus in their QoS broker on dynamic path conditions which influence the route between the source and destination involved in a bandwidth brokering scenario [2]. In other words, they manage admission control requests dynamically to reflect the changes occurring in a dynamic topology consisting of moving nodes. The brokering process is then responsible for path tracking for all hops that change in an end-to-end path. Such an approach is not scalable and ceases to perform well as the number of hops between the source and destination increases.

Nahrstedt et al. integrate the resource reservation functionality with their resource broker module [1]. We on the other hand rather use policing and discarding

traffic which violates the agreement between the served station and the broker on the last mile. Nevertheless, the approach used in [1] has several similarities to our resource management scheme. For instance, they use a client scheduler and client information table, a module on the terminal which communicates with entities in the network for optimizing admission control and resource utilization. We use a local resource management module on the client functionally similar to an operating system and is responsible for resource partitioning among traffic types; that is to say, creating flows out of one flow. This is outlined in Section IV of this paper.

Nahrstedt et al. also use event-based triggering to perform updates on the allocated resources by a broker to a terminal giving a dynamic touch to the brokering mechanism.

III. QoS BROKERAGE AND CAPACITY MANAGEMENT

A. *QoS budget*

As a result of the un-proportional fraction between overhead and data when the transmission speed of a WLAN station decreases with rate adaptations, the commonly used parameter "bit rate" does not provide accurate insight in load conditions and/or traffic demands in WLANs. The IEEE working group addresses the inefficiency by introducing block acknowledgements, in the draft IEEE802.11e standard. This will reduce, but not eliminate the situation. To deal with this, the solution proposed in [25], [26] is followed and a multi-service traffic profile is used as a throughput parameter. Each traffic profile consists of a tuple {Average Packet Size (APS), Average Packet Rate (APR)} and defines the QoS budget for each terminal. The QoS budget can further be distinguished into two levels, namely the *committed* and the *peak* budget. WLAN stations can determine autonomously whether their network consumption is kept within limits and/or whether a newly started network greedy application still fits within the QoS budget given. As defined in the IEEE802.11e standard, terminals may distinguish between various traffic classes based on the time sensitivity of each application and may store packets of each class in a separate queue that is served with different priorities to ensure the end-to-end priority to time sensitive connections. The QoS budget forms a SLA for the WLAN and dictates the size and the frequency of packets offered to the WLAN layer and it is up to the station to prioritize certain packets or not.

B. *QoS solution*

The shared nature of the wireless medium (in an unlicensed spectrum) poses several challenges and side conditions on QoS, including:

- Available resources may vary due to interference or unfavorable channel conditions of individual users and should be shared among all stations in an efficient manner, possibly taking into account the user's subscription.
- Resource consumption in the (wireless) access network should be predictable. This requires proper assignment and control of resources to each station.

- The admittance of stations should be limited so that an acceptable and configurable level can be realized.
- The solution should fit within the current QoS solutions offered by the IEEE 802.11e standard.

One aspect of QoS guarantees is to distinguish various priority classes. If the traffic ingress volume of a network exceeds the amount that can be processed, these mechanisms are not sufficient and QoS degradation is inevitable. This is why - besides priority - QoS solutions must include indications about volumes, specified in the QoS budget. A component named *QoS broker* is associated with one or several WLAN access points to determine whether new users can be added without jeopardizing the QoS budget assigned to other users.

Additional requirements to the resource planning in WLAN networks arise as stations may perform rate adaptation and may thereby affect the medium capacity and thus the available resources of all users associated. To counteract these and other unforeseen events (e.g. violations of the QoS budget), the QoS broker monitors the channel conditions and the user traffic to take measures when QoS guarantees may be jeopardized. Before users gain network access, their subscription must be obtained, defined in terms of QoS profiles. These profiles could either be stored locally, or in the network. In the latter case, vendor specific attributes of the authentication protocol are intercepted to obtain the subscription. From this set of QoS profiles, the QoS broker selects the one profile to be guaranteed and derives the minimum WLAN data rate needed for using this profile. The QoS profiles selected are subsequently communicated with the QoS element on the station. Consequently, the station is held accountable for not exceeding the selected QoS profile and is policed upon this limit. Traffic shaping prevents deletion of packets from each station. In turn, the QoS broker aims at meeting the QoS profile committed to each station by monitoring their traffic consumption as well as the channel conditions. A station moving away from the AP likely adapts its rate, consequently increasing the WLAN airtime consumption substantially. If a medium overload occurs, the QoS broker redistributes the network resources offered to all stations and updates the affected stations on their newly assigned QoS budget, see [26] for details.

IV. REAL-TIME TRAFFIC RESOURCE MANAGEMENT IN WLANs: THE PARAMETER INJECTION ALGORITHM

A. *Autonomic Payload Type Configuration*

Our algorithm which computes and allocates resources to real-time streams on the terminal requires information from several different input sources as seen in Figure 1:

- The average packet size and average packet rate from the QoS broker (described in Section II of this paper, on the residential gateway). The APS and APR together form the airtime allocated to a station and their product corresponds to the network resources a terminal is granted.

- The device profile signifying which media formats are supported by a terminal (CODECs in software and hardware)
- Information on the terminal from a local resource manager which determines what fraction of the overall network resources goes to each traffic class. The aim is to indicate how much of the overall bandwidth or airtime a station owns is allocated to real-time traffic.
- Estimated duration of stay of a mobile station within the vicinity of an access point without being subject to major reduction in assigned airtime. This is predicted using parameters such as motion speed of the terminal, coverage size (range) of the cell to be visited, and any other supplementary information supplied to the terminal via the Candidate Access Router Discovery (CARD) protocol [5] or other sources depending on the architecture used.

Figure 1: Collecting input and computing result for session update for injection into stream

According to RFC 3551 [19], one can see that the first column *s* labeled as “PT” corresponds to the payload type which is designated by a unique numeric code. This standardized notation is exactly the same one used within media object descriptors in SDP (Session Description Protocol), RFC 2327 [5]. Our aim is to automatically determine a suitable SDP configuration for various media channels and their corresponding parameters based on the dynamic situation within the network.

Figure 3 is an extension of the regular SDP payload type table with one added column on the right which depicts typical and statistically averaged data rates for various payload types. Swapping the table columns leads to what is seen in Figure 4.

Data Rate (Kbps)	Clock Rate (KHz)	Media Type	Encoding (PT)
64	8	PCMU	0
13.2	8	GSM	3
5.3, 6.3	8	G723	4
32	8	DVI4	5
64	16	DVI4	6
5.6	8	LPC	7

Figure 2: Table after swapping PT and data rate columns

Encoding (PT)	Media Type	Clock Rate (KHz)	Data Rate (Kbps)
0	PCMU	8	64
3	GSM	8	13.2
4	G723	8	5.3, 6.3
5	DVI4	8	32
6	DVI4	16	64
7	LPC	8	5.6

Figure 3: Data rates used by some formats and PT codes

Data Rate (Kbps)	Clock Rate (KHz)	Media Type	Encoding (PT)
64	8	PCMU	0
13.2	8	GSM	3
5.3, 6.3	8	G723	4
32	8	DVI4	5
64	16	DVI4	6
5.6	8	LPC	7

Figure 4: Reversing the PT table after extending it with the data rate column

This scheme performs a sequence of simple operations such as table column extensions, rotations, sorting, and row-merging. It interacts with a broker on the last mile in the residential domain. The broker allocates resources in the form of airtime budget periodically to the clients. Then an algorithm is needed to run on the client side and interact with the last mile broker in order to read the new budget and then manage it locally. Our solution performs autonomic resource management for different media channels belonging to real-time applications.

Bandwidth Range Audio Channel	Corresponding payload types and encodings	Filtering Stage (Profile dependent)	Final PT Types
0 - 8 Kbps	7(LPC), 4(G723[5.3/6.3]), 12(QCELP)	Each element in this column is a subset of its left neighbour; filtering based on device profile	7, 4, 12
8 - 16 Kbps	18(G729), 3(GSM)		18, 3
16-32 Kbps	15(G728), G726-16, G726-24		15
32-48 Kbps	5(DVI,8KHz), 16(DVI,11KHz), G726-32, G726-40		5, 16
48-64 Kbps	-		-
64-128 Kbps	8(PCMA),9(G722), 6(DVI,16KHz), 17(DVI,22KHz)		8, 9, 6, 17
> 128 Kbps	14 (MPA), 10/11 (L16)		

Figure 5: BW mapping to Audio PT Codes

The tables in Figure 5 show the mapping for audio; based on RFC 3551 [10], PT (Payload Type) values for various bandwidth ranges, is obtained from by aggregating several rows into ranges. Entries in the second column of the table are filtered based on the device profile which should contain a list of software and hardware-based supported formats and CODEC groups. For instance, some client gets assigned for the audio channel on the uplink 6 Kbps; therefore we end up in the 1st row of the table (Figure 5); within the 2nd column with the formats: LPC which corresponds to PT code 7, G723 PT which maps to code 4 and QCELP corresponding to PT code 12. Assuming the device profile supports only the LPC and QCELP CODECs, only SDP PT parameter values “7” and “12” will be assigned to the audio part of the media channel for the uplink for the particular node in question. Our algorithm then follows the same procedure for video media channels to configure the PT (payload type) codes for the SDP offer to be sent out.

Bandwidth Range for Video Channel	Corresponding payload types and encodings	Filtering Stage (Profile dependent)	Final PT Types
0-64 Kbps	34(H263Q CIF (352x288)), 32(H261Q CIF (352x288))	Each element in this column is a subset of its left neighbour; filtering based on device profile	34, 32
64-256 Kbps	34(H263 CIF (176x144)), 32(H261 CIF (176x144))		34, 32
>256 Kbps	26(JPEG), 32(MPV)		26

Figure 6: Video PT computation table

Remark: our ongoing research also attempts at refining the bandwidth or data rate range bounds for finer granularity and better performance, especially concerning video encodings and formats.

B. Parameter Injection Process

The session update part consists of a computation phase and an injection phase outlined in Figure 7. The injection phase itself corresponds to calling a class with the newly computed parameters for reassigning the session parameters to obtain a new RTP stream with the updated configuration. This dynamic feature is supported already in many SIP stacks and media frameworks, but still may be subject to buggy stack performance in some cases.

Moreover, the availability of protocols such as CARD [5], which provide information in a proactive fashion and the availability of information on coverage, encourage us to profit from this type of knowledge to reduce session signaling delay. We also try to configure the session and perform an update on it automatically and then inject the new calculated parameters into the RTP stream using the stack on the client which is an end point of the stream. In other words, terminals which use SIP-based real-time applications shall exchange their complete list of RTP supported payload types, once at the beginning. This delivered information is enough to determine the next

session configuration when one endpoint is subject to changing channel conditions or a handover or a resource update.

Figure 7: Parameter injection procedure

The preconditions for performing a direct injection of parameters for RTP into a running stream are listed below:

- *Availability of enough information* to compute realistic session parameters for the update; this is available thanks to certain entities within the architecture, located on the client (Figure 1) and on the last mile (QoS Broker mentioned in Chapter III).
- *Standard conformity*; this has been investigated and it is indeed the case. As stated in RFC 3264 [8] by Rosenberg and Schulzrinne: “An Offer/Answer Model Session Description Protocol”, section 5.1 on Unicast Streams. The related statement within the RFC is depicted below.
- *Feasibility*; current RTP protocol implementations, SIP stacks, and signaling frameworks can cope with this. This is also the case, as we were able to successfully conduct parameter injection into running RTP streams bypassing renegotiation with minor difficulties.

Quoting RFC 3264 [8]: An Offer/Answer Model Session Description Protocol Section 5.1 “Uni-cast Streams”:

The list of media formats for each media stream conveys two pieces of information, namely the set of formats (codecs and any parameters associated with the codec, in the case of RTP) that the offerer is capable of sending and/or receiving (depending on the direction attributes), and the RTP payload type numbers used to identify those formats. ***If multiple formats are listed, it means that the offerer is capable of making use of any of those formats during the session. In other words, the answerer MAY change formats in the middle of the session, making use of any of the formats listed, without sending a new offer.***

Figure 7 shows that negotiation has to take place at the beginning. It is extremely important that not only parameters for a particular session start are exchanged,

but rather that all capabilities of both end-points are shared within the first SDP frame as shown in the picture. Then during mobility, having the full list of each others' capabilities, terminals would not need to negotiate for a new session update because the resource information is available from the QoS broker and the full Payload Type (PT) and CODEC list is obtained in the first cycle. Furthermore, other information is obtained by local modules on the terminal (Figure 1). The process of updating a session upon resource reallocation or a handover is thus composed of 2 main phases: a computation and information gathering phase followed by the phase where the computed parameters are then injected into the running RTP stream (Figure 7).

Concerning the computation of the data rate for an individual SIP-based (real-time or RTP based session), the formula in Figure 8 is used. Simply multiplying the average packet size by the average packet rate yields the effective airtime data rate. Then this has to be multiplied by 8 to convert bytes to bits. The result has then to be multiplied by the fraction of overall bandwidth which is allocated to real-time traffic; this corresponds to the fraction (percent) variable divided by 100. Finally, the intermediate result which is the data rate allocated to real-time traffic has to be divided by the number of multimedia applications. The division by a factor of 1000 is to obtain a result in Kbps not bps. Then we assign one fourth of the bandwidth to the audio channel and the rest to the video channel within a single multimedia application.

$$\begin{aligned} \text{Datarate} &= (\text{apr} * \text{aps} * 8 * \text{fraction}) / \\ & \quad (100 * \text{appCount} * 1000); \\ \text{audioDR} &= \text{datarate} / 4; \\ \text{videoDR} &= 3 * \text{datarate} / 4 \end{aligned}$$

Figure 8: Computation process

The next section will evaluate the performance of the implemented algorithm and provide some insight on limitations and gains achieved.

V. PERFORMANCE RESULTS AND EVALUATION

A. Gain on Session Delay Reduction

Figure 9 depicts the pure case of comparing the 2 approaches with the same scenario (same input file with network parameters, same network conditions, same duration for each session before update), and then the times required to set up the sessions are marked (based on the signaling procedure in Figure 7 versus classical negotiation). The negotiation algorithm interacts more with remote entities, whereas the injection scheme requires a slightly longer local computation but fewer remote interactions.

Figure 9: delay profiles of negotiation versus injection algorithm

Based on figures 9, 10, and 11, we make the following observations:

- Convergence towards a statistical pattern as verified with many runs is observed in Figure 10 whereby the session signaling delay average stabilizes at around 38 milliseconds when using the parameter injection algorithm. Session signaling reduction by as much as a factor of 2.5 is achieved for small runs with 300 handovers or parameter update cycles and a factor of 94 milliseconds/38 milliseconds = 2.47 times is achieved for large runs which is still a significant reduction on session signaling delay.
- Figure 11 shows that the jitter profile, which does not vary from large to small runs, has a low jitter value on the average of around 10 to 15 milliseconds. This is very good for real-time traffic especially for the jitter-sensitive voice traffic category.

Figure 10: Parameter injection performance

B. Gain in Terms of Reduced and Stabilized Jitter

Figure 11: Jitter profile for the injection algorithm

C. General Benchmarking

Parameter/Level	Good	Acceptable	Poor
Delay	0-150 msec	150-300 msec	> 300 msec
Jitter	0-20 msec	20-50 msec	> 50 msec
Packet Loss Rate	0-0.5%	0.5-1.5%	> 1.5%

Table 1: Quality levels and parameters from [24]

According to [24], values for delay, jitter and packet loss rate (basically network QoS atomic parameters) can be grouped into 3 main ranges for quality good, acceptable and poor. The negotiation algorithm falls on the border between good and acceptable in terms of delays under good network conditions and in terms of jitter, it falls within the acceptable range, being subject to distortions and perturbations for VoIP users in sensitive wireless environments. On the other hand, the injection approach proposed provides very low jitter values and also drastically shorter session update times which never even get out of the first half of the good interval.

VI. CONCLUSION

As seen throughout the document, the implementation of the algorithm (injection algorithm) which is a ramification case of the parameter negotiation RFC saves greatly on performance, reduces delays and provides better (lower) jitter values as well as session update delay values. Concrete verification of feasibility, implementation technical details and concrete performance results in real scenarios have been presented as well. We notice that we achieve an improved value of 10 to 15 milliseconds for jitter and a factor of 2.47 in terms of session signaling delay. Future work would be to further refine the different ranges where PTs are mapped to data rate intervals and also to optimize work on the terminal, especially in terms of partitioning bandwidth among traffic types and among several applications within a single traffic class. Also issues such as dependency on the topology and on traffic flow patterns would be beneficial for the algorithm.

ACKNOWLEDGMENT

The authors would like to thank Frans Panken from Alcatel-Lucent Bell Labs Europe for his valuable suggestions and comments.

References

- [1]. K. Kim and K. Nahrstedt, "A Resource Broker Model with Integrated Reservation Scheme," in Proceedings of the IEEE International Conference on Multimedia and Expo. New York, NY, august 2000
- [2]. A. Krishnamurthy, L. Qian, Y. Wang, P. Dauchy, and A. Conte, "A New Coordinated Scheduling Algorithm in Distributed Bandwidth Broker QoS Architecture," in IEEE Communications May 2005, vol. 1, pp 384-388. Proceedings of the IEEE International Conference on Communications, Seoul, Korea
- [3]. Z. Duan, Z. Zhang, Y. Hou, and L. Gao, "Core Stateless Bandwidth Broker Architecture for Scalable Support of Guaranteed Services," IEEE Transactions on Parallel and Distributed Systems, Vol. 15, No., 2 pp.167-182, February 2004
- [4]. Z. Zhang, Z. Duan, and Y. Hou, "On Scalable Network Resource Management Using Bandwidth Brokers," IEEE Network Operations and Management Symposium, IEEE/IFIP April 2002
- [5]. M. Liebsch, A. Singh, H. Chaskar, D. Funato, and E. Shim "Candidate Access Router Discovery (CARD)," Request for Comments 4066, IETF site: www.ietf.org/rfc/rfc4066.txt
- [6]. IETF RFC 2327: SDP: Session Description Protocol, by Handley and Jacobson, April 1998, IETF Drafts
- [7]. IETF RFC pages: <http://www.ietf.org/rfc.html>
- [8]. IETF RFC 3264: An Offer/Answer Model with Session Description Protocol (SDP); J. Rosenberg and H. Schulzrinne, 2002
- [9]. B. Girod, J. Chakareski, M. Kalman, Y. J. Liang, E. Setton, and R. Zhang; Advances in Network-adaptive Video Streaming; Information Systems Laboratory, Department of Electrical Engineering Stanford University, Stanford CA 94305, USA
- [10]. H. Schulzrinne; A Streaming Architecture for Next Generation Internet Ashutosh Dutta Telcordia Technologies, 445 South Street, Morristown, NJ 07960 Computer Science Department, Columbia University, New York, NY 10027
- [11]. T. Truetti, S. Parisi, and J. Bolot, "Experiments with a layered transmission scheme over the internet", Technical Report RR-3296, INRIA, France
- [12]. R. Rejaie, M. Handley, and D. Estrin, "An end-to-end rate-based congestion control mechanism for real-time streams in the internet". INFOCOMM 99 Proceedings, 1999.
- [13]. S. Floyd, M. Handley, J. Padhye, and J. Widmer; "Equation-based congestion control for unicast applications: the extended version", International Computer Science Institute technical report TR-00-03, March 2000.
- [14]. IEEE 802.11 official group site: grouper.ieee.org/groups/802/11/
- [15]. IETF RFC 3261: SIP: Session Initiation Protocol, by Rosenberg, Schulzrinne, Camarillo, Johnston, Peterson, Sparks, Handley, and Schooler; June 2002
- [16]. Internet Engineering Task Force (IETF): <http://www.ietf.org>
- [17]. IETF RFC 3558 - RTP Payload Format for Enhanced Variable Rate Codes (EVRC) and Selectable Mode Vocoders (SMV)
- [18]. IETF RFC 3312: Integration of Resource Management and Session Initiation Protocol (SIP); G. Camarillo, W. Marshall, and J. Rosenberg; October 2002
- [19]. IETF RFC 3551: RTP Profile for Audio and Video Conferences with Minimal Control; H. Schulzrinne and S. Casner, July 2003 [Obsoletes RFC 1890]
- [20]. A. M. Al-Naamany and H. Bourdouce. Fuzzy-Logic-Based TCP Congestion Control System; Network control and engineering for QoS, security and mobility II table of contents Pages: 180 - 190 Year of Publication: 2003 ISBN:1-4020-7616-9
- [21]. IETF RFC 2748: Common Open Policy Service Protocol
- [22]. D. Durham, J. Boyle, R. Cohen, S. Herzog, R. Rajan, and A. Sastry; June 2000
- [23]. Willamette Integrated Technology Services, on resources and bandwidth usage: www.willamette.edu/wits/resources/netiquette/bandwidth.htm
- [24]. P. Calyam and C. Lee: Characterizing Voice and Video Traffic Behaviour over the Internet: International Symposium on Computer and Information Sciences (ISCIS). Proceedings published by Imperial College Press in a special edition of "Advances in Computer Science and Engineering" Book Series, 2005
- [25]. F.J.M. Panken, G.J.Hoekstra: Multi-service traffic profiles to realise and maintain QoS guarantees in wireless LANs, submitted for possible publication in IEEE Wireless.
- [26]. Frans Panken, Gerard Hoekstra, Sietse van der Gaast: Resource allocation and guarantees for real-time applications in WLANs, Teletronikk, Telenor's journal of technology, Volume 102 No. 3/4 2006; ISSN 0085-7130