QOS FRAMEWORK FOR WIRELESS NETWORKS

Anna Lina Ruscelli
Scuola Superiore Sant'Anna- Retis Lab
Vai G. Moruzzi, 1, Pisa

Gabriele Cecchetti
Scuola Superiore Sant'Anna- Retis Lab
Vai G. Moruzzi, 1, Pisa

ABSTRACT

This paper presents a work in progress about a cross-layer approach to support Quality of Service for wireless multimedia applications, building a suitable framework over the top of the heterogeneous wireless MACs. It lets to enhance the existing QoS support provided by standard MAC protocols and it uses the contract model to guarantee QoS, taking into account the applications requests. It negotiates dynamically *Application Level Contracts* which will be translated seamlessly in *Resource Level Contracts* for the underlying network services from which it receives the feedback to adjust the scheduling algorithms and policies to provide soft guarantees. The framework comprises QoS Manager, Admission Control, Enhanced Scheduler and Feedback System. The QoS manager component is a middleware able to dynamically manage available resources under different load conditions in a transparent manner to application level.

KEYWORDS

QoS, framework, middleware, wireless networks.

1. INTRODUCTION

The increasing spreading of multimedia applications in wireless communication networks has produced a growing interest in Quality of Service (QoS) support and several studies have proposed possible solutions. Real-time services with strict latency/throughput requirements (e.g. Voice over IP, video conference, audio and video streaming) require guarantees for constant bit rate (CBR) and variable bit rate (VBR) traffic streams (TS). Moreover, wireless networks involve space and time varying characteristics, differently from what it happens for the wired ones and they are subject to fast changes in Signal to Interference plus Noise Ratio (SINR) due to phenomena like path loss, shadowing, multipath fading, signal attenuation and interference. In this context one trend is to use an adaptive QoS system with a relative QoS differentiation [Dovrolis2002], based on different priority classes of Differentiated Service architecture to deliver multimedia data [Shin2001][Patmanabhan1999]. Another remarkable point of view is to introduce a cross-layer design with adaptive QoS assurance for multimedia transmissions [Alwin1996][Mirhakkak2001].

In this article we present a novel framework to provide a comprehensive soft QoS support for multimedia traffic streams. Our approach is inspired both to cross-layer architecture idea and QoS differentiation and it allows to interface seamlessly multimedia applications with lower layers of a wireless network. We specifically focus on our wireless application framework components: *QoS manager*, *Admission Controller*, *Scheduler* and *Feedback mechanism*, with particular attention to QoS manager witch has the role of a middleware between various multimedia applications, from which it accepts different QoS requests, and lower network layers, translating these requests in the specifics of each involved medium access standard, handling time-varying network conditions, heterogeneous traffic streams and managing efficiently link layer resources.

2. APPLICATION FRAMEWORK

The framework has a cross-layer architecture composed by a middleware for QoS and the scheduling subsystem. The former transparently manages the communication levels for applications while the latter consists of some building blocks to regulate the medium access. This subsystem may vary depending on the particular MAC(s) used (e.g. IEEE 802.11, IEEE802.16, mobile public networks, wired networks), whereas the QoS manager is independent both to application and MAC, to let one possible solution to convergence question (e.g. 4G communications [Shelper2005][Lu2006]). The framework implements a contract based scheduling that is suitable to integrate QoS support provided by different communication standards. The contract model lets to the applications to dynamically negotiate their own set of complex and flexible execution requirements with QoS manager which acts as a proxy between the applications and underlying network layers. Such approach represents high level abstraction that lets practitioners to concentrate on the specification of the application requirements.

2.1 QoS Manager

QoS manager [Cucinotta2005] is the middleware layer that mediates between software applications and underlying components, i.e. heterogeneous networks MAC layer, providing appropriate interfaces. It translates the high level QoS requirements of the application into transmission parameters values, then it negotiates them with admission control. From the application perspective, the requirements of an application component are specified by a set of *Application Level Contracts* (ALC), which are negotiated with the underlying implementation. From the scheduling subsystem perspective the requirements are expressed by a set of *Resource Level Contracts* (RLC) based on available network resources. QoS manager has to check, as part of the negotiation, if there are enough resources to guarantee all the specified minimum requirements that is needed to fulfill the ALC, while keeping guarantees on all the previously accepted contracts negotiated by other application components. Eventually it adapts the requirements to available resources, adapting dynamically the resource allocation in order to optimize the resource utilization without sacrificing on QoS requirements. If a result of this negotiation is successful, the scheduling subsystem reserves enough capacity to guarantee the minimum requested requirements and it reclaims any spare capacity fairly redistributing among different traffic flows requesting additional resources. Moreover, when an application wants to change the contract profile, QoS Manager performs a so-called adaptive resource allocation: it contacts again the corresponding admission control service and negotiates a new RLC, tuning service parameters. Finally, in the case an overload occurs (e.g. due to varying network conditions or if a more important QoS request is received), it can decide to change one or more ALCs to degrade the QoS level of one or more applications by a call-back notification so that the application itself can adapt its QoS requirements. The QoS manager is a "*two-side*" Application Program Interface (API) located as a middleware on network nodes. The upper side interfaces applications while the bottom side interacts with scheduling subsystem. The applications can call the following functions of QoS manager. `int request_ALC (struct *ALCspec)` is used by applications to negotiate an ALC with QoS manager. The latter checks the requirements specified with `struct *ALCspec` through *Admission Control subsystem* and returns the result 0 if the ALC is accepted, 1 if the ALC is modified or 2 if the ALC is rejected. At any time an application can request to modify the ALC by `int modify_ALC (struct *ALCspec)` which behaves like `request_ALC`. It can cancel the current contract issuing `int cancel_ALC (struct *ALCspec)`, which should return `0` unless an error is occurred and that is used also to abort a contract. The API bottom side is used by QoS manager to interact with the undelying levels. It consists of three functions for every protocol managed by QoS manager. The function `int request_RLC_`*`proto`* `(struct *RLCspec)`, where proto may be any supported network protocol, is called by QoS manager to negotiate a RLC with the scheduling subsystem and it returns `0` if the RLC is accepted, `1` if the RLC is modified or `2` if the RLC is rejected. If the RLC is modified and ALC can still be satisfied, the QoS manager adjusts `ALCspec` and return it to the application requesting the corresponding contract, while if the RLC is modified but the ALC cannot be satisfied the QoS manager cancels the RLC through the `int cancel_RLC_`*`proto`* `(struct *RLCspec)` function and then notifies to the application that it cannot accept the ALC. RLCs are also cancelled when ALC expires. The function `int get_RU_`*`proto`* `(struct *RU_`*`proto`*`)` is used by QoS manager to query the scheduling

subsystem about protocol resource utilization. The returning `RU_proto` lets the QoS manager to enhance its negotiation capability.

2.2 The Scheduling Subsystem

Admission control verifies if there are sufficient resources for medium access to satisfy QoS manager requests. It computes the theoretical new bandwidth utilization and it checks if it is admissible without degradation of preexistent transmissions. The response is sent back to the QoS manager. If the instance request is successful a RLC is established and the QoS manager can communicate transmission parameters to the corresponding scheduler. The general admission test used is: $\sum_{i=1}^{N} \frac{Q_i}{P_i} \leq U_{\text{lub}}$ where $Q_i \equiv C_i/r_i$ is the average time budget of the medium which is reserved to the i^{th} network station transmitting within each period P_i; r_i is the physical bit rate assumed for admission control computations of the i^{th} traffic stream TS_i, C_i are the bytes transmitted during the P_i and U_{lub} is least upper bound utilization factor computed for the worst-case available bandwidth. If there is not enough bandwidth to serve the new request, three different admission control policies exist which act as follows. With *Saturation policy* the highest possible budget is assigned to the task so that the total resource utilization does not exceed U_{lub}, whereas with *compression policy*, in respect of the established ALCs, all the RLCs are recomputed ("compressed") so that we can make new space for the new request. Finally with *reject policy* the transmission is rejected.

The **scheduler** manages each TS transmission for each admitted flow and it assigns dynamically both the period P_i and transmission duration TX_i to follow the channel variability and streams characteristics. We propose a scheduler which can handle TS with soft Real Time guarantees [Stankovic1998] with special regard to VBR flows which are supported by assigning transmission duration in agreement to the effective temporal demands. The assignment of P_i is dynamic, so it lets to increase the transmission frequency of the applications having in queue traffic with tightening requirements of QoS. The scheduler is also able to reclaim the unused time of nodes which have exhausted their transmission before the end of their transmission duration and then it assigns that time to the nodes which have still useful data to transmit. Delay or advance of the transmission with respect to the pre-agreed rate (in terms of bytes which have been anticipatively used or have not been transmitted by node) are formalized as the scheduling error $\varepsilon_i^{(k)}$, defined, at the k^{th} time instant, as the difference between the cumulated bytes to transmit $z_i^{(k)} \equiv kC_i^{(k)}$ and the bytes actually transmitted $z_i^{(k)}$: $\varepsilon_i^{(k)} \equiv z_i^{(k)} - \bar{z}_i^{(k)}$ The dynamic equation for the evolution of the scheduling error for the i^{th} real-time data flow is: $\varepsilon_i^{(k+1)} \equiv \varepsilon_i^{(k)} + C_i^{(k)} - \gamma_i^{(k)} Q_i^{(k)}$ where $\gamma_i^{(k)}$ is the actual channel speed.

The **feedback** mechanism senses the effective information acknowledged by nodes to vary transmission parameters of the scheduler in order to minimize the scheduling error. The feedback system can compensate little variations of network conditions without the intervention of admission control to establish new RLCs. During normal condition, if $\sum_{i=1}^{N} \frac{Q_i^{(k)}}{P_i} \leq U_{\text{lub}}$, feedback system controls the scheduling error assigning: $\forall i, Q_i^{(k)} \equiv \tilde{Q}_i^{(k)} = \frac{C_i^{(k)} + \alpha_i \varepsilon_i^{(k)}}{\rho_i^{(k)}}$ where $\tilde{Q}_i^{(k)}$ is the required assigned budget to compensate the scheduling error, $\alpha_i \in]0,1]$ is a fraction of the current scheduling error for each TS_i and $\rho_i^{(k)}$ is the channel speed at the physical layer.

2.3 Framework Emulated Environment

Currently we are developing a demonstrator of the whole framework with *Network Simulator 2* software package. We are extending the support offered by 802.11e component adding a new enhanced scheduler which better supports CBR and VBR TSs. A new library which implements the QoS manager is built. The application level consists of several sources of generated traffic with strict latency/throughput requirements (e.g. Voice over IP, video conference, audio and video streaming). In Fig. 1 we evaluate a scenario with up to six stations with VoIP or videoconference traffic (VC) and we report the data throughput reached by the stations with data traffic. If there are not any stations with CBR or VBR TSs, the data throughput is maximum and the framework behaves in a very similar way to the IEEE 802.11e reference scheduler.

Otherwise, the reduced overhead of the adopted scheduler improves the throughput achievable by data stations. Fig. 2 shows the delay of uplink and downlink VoIP and VC traffics versus number of stations .

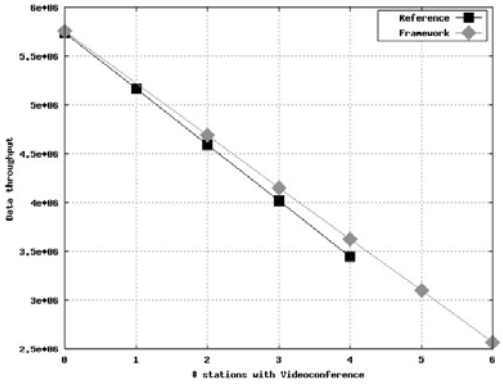

Figure 1. Data throughput of unreserved HCCA capacity

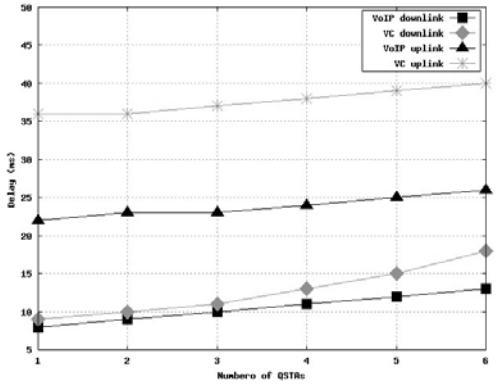

Figure 2. Access delay vs. number of stations

3. CONCLUSION

In this paper we have presented a brief description of a work in progress about a cross layer framework to integrate QoS support for wireless networks under time varying network conditions and different traffic specifications. It provides an interface to QoS support mechanisms for any applications with tightening guarantees and temporal boundaries, allowing applications establish contracts with QoS manager that administers the available resources from underlying subsystems and acts as a proxy towards different network subsystems which manage different wireless network protocols. We propose a subsystem scheduler that supports real-time applications, variable packet size and variable bit rate traffic streams, showing that it is suitable to be used by software requesting application level contracts, and it is able to manage available resources dynamically under different load conditions. Feedback mechanism tunes the scheduler behavior during transmissions. Currently we are developing a simulated environment of this framework through *ns-2* software. The framework is applied to recent IEEE 802.11e WLANs and preliminary results are shown.

ACKNOWLEDGEMENT

This work has been supported by FRESCOR EU project (Contract n. 034026).

REFERENCES

[Dovrolis2002] Dovrolis, C. et al, 2002. Proportional differentiated services: Delay differentiation and packet scheduling. *In IEEE/ACM Trans. Networking*, Vol. 10, Feb. 2002,pp 12-26.

[Shin2001] Shin, J. et al, 2001. Quality-of-service mapping mechanism for packet video in diffentiated services network. *In IEEE Trans. Multimedia*, Vol. 3, June 2001,pp. 219-231.

[Patmanabhan1999] Patmanabhan,V., 1999. Using differentiated services mechanism to improve network protocol and application performance. *Proceedings of IEEE RTAS Workshop QoS Support for Real-Time Internet Applications*.

[Alwin1996] Alwin, A. et al, 1996. Adaptive mobile multimedia networks. *In IEEE Pers. Commun.*, Vol. 3, No. 2

[Mirhakkak2001] Mirhakkak, M. et al, 2001. Dynamic bandwidth management and adaptive application for a variable bandwidth wireless environment. *In IEEE J. Select. Areas Communn.*, Vol. 19, Oct. 2001, pp 1984-1997.

[Lu2006] Lu, W. and Hu, J., 2006. Open wireless architecture – the core to 4G mobile comunications. *In CIC Journal of Vhina Communications*, Apr. 2006.

[Stankovic1998] Stankovic, J. et al., 1998. *Deadline scheduling for real-time systems*. K. A. Publishers, Boston.

[Cucinotta2005] Cucinotta, T. et al., 2005. QoS management through adaptive reservations. *In Real-Time Systems Journal*, Vol. 29, no. 2-3, Mar 2005.