Alternative Traffic Conditioning Mechanisms for the Virtual Wire Per-Domain Behavior

Uwe Walter, Klaus Wehrle, Martina Zitterbart

***Abstract*— Interactive networked multimedia applications are sensitive to delay and jitter affecting the transmission of their data packets. Without the support of quality of service, the nature of the Internet makes it difficult to deliver data streams in the demanded constant and steady way these applications need for an acceptable performance. Because of this limitation, value-added services like the "Virtual Wire"-Per-Domain-Behavior (VW-PDB) impose the obligation of traffic conditioning mechanisms in the transmission path to assure a continuous data stream to an end user. However, each traffic conditioning action is a potential source for additional delays experienced by the processed data packets.**

In this paper we will show, that the configuration suggested by the VW-PDB for traffic conditioners in a last hop router, can quickly lead to unwanted significant additional delays. Although applications will then be served with a constant data stream with minimal jitter, the traffic conditioning might exceed their delay limitiations. Since modern multimedia applications normally use their own playback buffers for jitter compensation, they do not depend on a complete jitter-free transmission, like the VW-PDB tries to establish by emulating a dedicated switched-circuit. Hence, two suggestions with more flexible traffic conditioning mechanisms in the last-hop-router will be presented. Both approaches will be compared to the strict rules prescribed by the VW-PDB in terms of jitter compensation and additional delay.

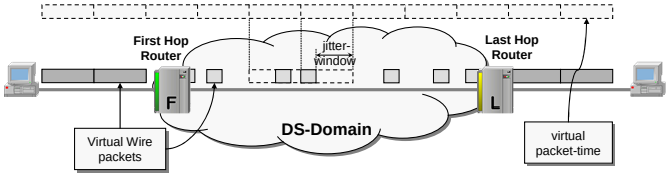

Fig. 1. Conceptual model of a virtual leased line using the Virtual Wire PDB

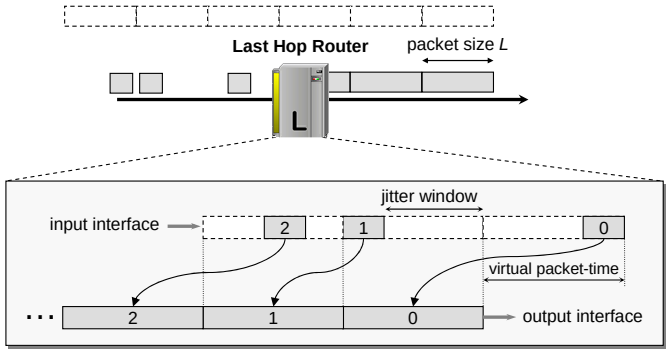

Fig. 2. Delay of packets in the last hop router for smoothing their transmission rate to bandwidth R of the virtual leased line

I. Introduction

The development of the Virtual Wire Per-Domain Behaviour [6] was aimed to allow the provisioning of an end-to-end service with a quality similar to a *fixed switched circuit* (*virtual leased line*). In terms of quality-of-service (QoS) this means that, on one hand there is a fixed bandwidth R which is ensured in a deterministic way (no packet losses) and on the other hand the end-to-end delay and jitter are minimized and bounded each by a guaranteed limit.

Consequently, from the perspective of two end systems, the Virtual Wire Per-Domain-Behaviour (*VW-PDB*) can be seen as the substitution of a dedicated leased line with bandwidth R (cf. fig. 1). Thus, the VW-PDB could fulfill the requirements of applications whose data could otherwise only be transmitted over circuit-based networks, e.g. telephone calls, video conferences and other real-time applications [6]. Because Virtual Wire is at the moment the only proposed Diffserv-PDB offering an adequate support for such non-adaptive real-time applications, its eligibility for these usage classes will be examined in the following.

It will be shown that the VW-PDB has several attributes that are essential to the guarantee of a worst-case maximum jitter bound, but are opposed to the demand of the lowest possible delay. Moreover, to be able to ensure such a maximum jitter limit, certain strict restrictions about the bandwidth that can be used by VW, must be fulfilled. Because most non-adaptive applications do not need such a strict guarantee of the maximum jitter, two possible modifications will be proposed in section III. They are more orientated at the requirements of interactive real-time applications with the ability to compensate a certain amount of delay variation.

Institute of Telematics, University of Karlsruhe (TH), 76128 Karlsruhe, Germany, (email: {walter,wehrle,zit}@tm.uka.de)

II. Traffic Conditioning in the Last Hop Router

The main objective target of the Virtual Wire PDB is the emulation of a virtual leased line of a configured bandwidth R. Consequently, two consecutive packets may only arrive at the receiver with this maximum rate R. However, due to the nature of the Internet, it is possible that an originally constant VW-data-stream gets disturbed, the packets are displaced and lose their smoothed characteristic. This may even happen in quality-of-service supporting networks, e. g. DiffServ domains, due to various effects caused by aggregation and disaggregation of several flows using the same network service.

If all access control conditions of the VW-PDB [6] are met, the delay variations are bounded within the jitter window and can be compensated and evened out by the last hop router.

The maximum jitter can accrue, if one packet experiences the least possible transmission delay (and is therefore placed on

the left most position in its jitter window) and the consecutive packet needs the longest transmission time – what equals being placed on the right most position of its jitter window (cf. fig. 1). To be able to deliver these packets to the receiver with no more than rate R, it is proposed in the VW specification [6] to delay the first packet in the last-hop router by one virtual packet time $P_R = \frac{L}{R}$, where L represents the maximum length a packet (cf. fig. 2).

Consequently all variations in the transmission time up to the maximum allowed value of one jitter window can be compensated and the receiver will not be able to notice a difference to the use of a leased line of bandwidth R. However, this buffering for one virtual packet time causes a (sometimes significant) additional delay of the Virtual Wire packets. Unfortunately, this is extremely undesirable, since especially applications using the VW-PDB are naturally intolerant to delay, as they tend to belong to the interactive real-time class and need the quickest possible transmission. Above all, it is normally not necessary to delay the first (and all following) packets for a whole virtual packet time to compensate the maximum jitter, as it will be shown by simulative experiments presented in section III-A.

A. Evaluation of the Virtual Wire PDB

The Virtual Wire PDB as proposed in [6] is able to emulate a virtual leased line, but only when some conditions concerning the maximum size of the VW aggregate $\overline{R}$ and the allocation of bandwidth to seperate VW data flows are met. These limitations ensure that the maximum delay variations are bounded inside a jitter window and can therefore be compensated by the last-hop router by buffering the packets for one virtual packet time.

It is possible to specify a deterministic upper bound for the end-to-end delay and jitter, which can always be guaranteed. However, this strong guarantee demands rigorous restrictions [6]:

- To limit the delay variations to the size of one jitter-window, only a certain fraction of the available bandwidth may be used for the Virtual Wire PDB that will mostly be smaller than 50%.
- Due to possible aggregation effects, the bandwidth fraction of a link that may be used by a Virtual Wire data stream j is limited to $\frac{1}{1+\max\{n_j\}}$, if this data flow might cross n_j other VW streams on its way through the Diff-Serv network. Especially since the Virtual Wire PDB is supposed to be used by applications with quite different bandwidth requirements, for example Voice-over-IP phone calls at 64 kbps or video streaming at 1.4 Mbps, this implies a serious limitation.

Consequently, it will hardly be profitable for a network provider to introduce a service with such restrictive limitations and likely a minor user basis. Furthermore, the VW-PDB does not really offer the best possible quality guarantees demanded by interactive real-time applications. Though delay and jitter both are limited by upper bounds, the main claim calls for the lowest possible delay, that Virtual Wire can not always fulfill, due to the extra delay for jitter compensation in the last-hop router.

The inspection of a 64 kbps packet stream reveals that it experiences an additional delay for jitter compensation in the last hop router and it may get delayed even more, when it is subject to traffic shaping in the first hop router. It will be buffered for a whole virtual packet time $P_{64\,\mathrm{kbps}}$. Assuming the packet lengths equals 500 Byte[1] this results in a duration of $\frac{(8{\cdot}500)\,Bit}{64\,kbps} = 62.5$ ms. Such an additional delay is hardly tolerable, above all by an interactive real-time application [2]. Figure 3 shows the resulting additional delays caused by the Virtual Wire buffering in the last-hop router for different packet sizes and bandwidths as proposed in [6].

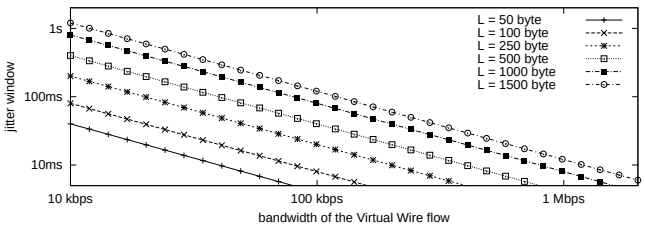

Fig. 3. Additional delay by the jitter window from buffering of Virtual Wire in the last-hop router

As we determined from the results of many simulation runs in [9], the specified buffering of the VW-PDB is too long, above all for the majority of packets from a VW data stream and only adds an unnecessary, sometimes even disturbing, additional delay. Hence, two alternative approaches will be presented in the following, that partially shift complexity from the net to the applications, according to the Internet's „end-to-end" desgin principle [7]. These options give the user more flexibility and above all make it possible for him to choose an alternative VW realization with a much smaller delay.

III. Alternative traffic conditioning mechanisms for the VW-PDB

In section 1 it was already mentioned that on the one hand the restrictions for its deployment are high and on the other hand the additional delay from the buffering in the last-hop router contradicts the demands of interactive real-time applications. In the following it will be analyzed, what the maximum needed delay will be at all. Based on this results, two possible modifications will be proposed making the Virtual Wire PDB more attractive to the class of delay sensitive realtime applications.

A. The problem of the unknown delay of the first packet

To gain enough buffer to smooth a VW data stream, the traffic conditioner in the last-hop router, delays the first packet of an arriving flow for the duration of a virtual packet time $P_R = \frac{L}{R}$. This duration corresponds to the maximum guaranteed jitter, a VW packet may experience under the conditions described in section II-A and [6]. In the scope of this paper, a lot of simulative examinations have been conducted within [9], that show it is necessary to respect these conditions to be able to meet the maximum jitter limits.

[1] 500 Byte is chosen as a mean value, as packet sizes vary greatly between different realtime applications, ranging from about 80 Byte for a voice only NetMeeting transmission (G.723), 566 Byte for a LiveLAN session ([8]) up to over 1000 Byte for NetMeeting with activated video transmission.

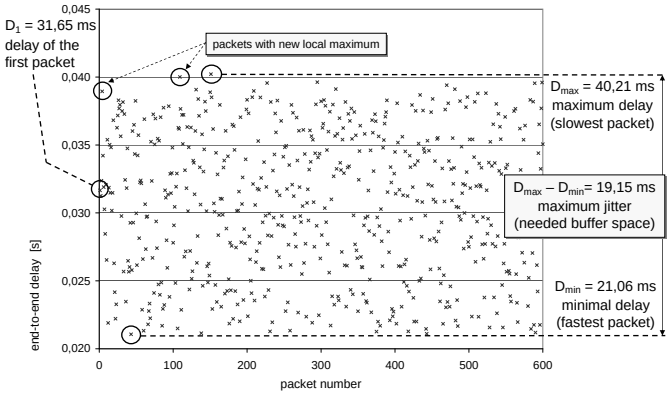

Fig. 4. Exemplary plot of observed delay of a typical Virtual Wire data flow between the first-hop and last-hop router

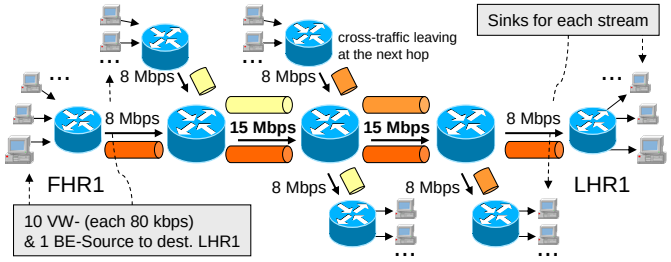

Fig. 5. Configuration example of one simulation network

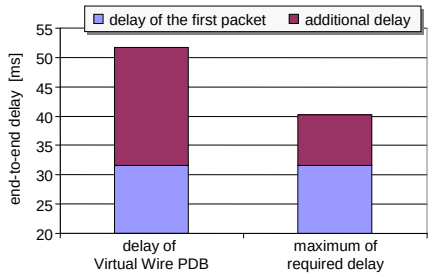

Fig. 6. Maximum end-to-end delay of the Virtual Wire data flow from figure 4

However, to restore a smooth data stream in the last-hop router it is not always necessary to buffer the first packet for a full amount of one virtual packet-time. This is implied by the fact that the maximum needed buffering depends on the delay of the first packet. The arrival of the first packet represents a kind of reference point for the receiver, to which it synchronizes itself to the rate R, measures the delay variations of consecutive packets and compensates them with buffering.

Only in case the first packet would experience the least occurred delay, the needed buffering will indeed be as long as a whole virtual packet time. To demonstrate this behaviour even better, the actually needed delay will be shown in an example. For this purpose, figure 4 depicts the delay progression of the first 600 packets of an example VW data flow. These delays have been measured in a simulation network depicted in figure 5 between the first-hop router FHR1 and the arrival at the last-hop router LHR1 (note: in this context of alternative traffic conditioning mechanisms only the very last DiffServ-capable router on the path to the destination is called last hop router, so no domain boundaries are considered). Other simulation runs performed in [9] have all shown similar behavior.

In the presented example, the first VW packet needed $D_1 = 31.65\,\text{ms}$ to traverse the network. If it would be forwarded immediately – *without* any buffering – a constantly smooth packet stream could only be assured when the delays of *all* following packets would be *at most* as high as the delay D_1 of the first packet. Hence, it is necessary to buffer the first packet as long as needed to be able to compensate the delay of all following packets that traverse the network slower. The minimal buffering that is absolutely necessary can be theoretically quantified by calculating the difference of the delay of the slowest packet D_{max} and the delay of the first packet D_1 from the regarded data flow. In the example in figure 4 the delay of the first packet was D_1 =31.65 ms and the highest delay was D_{max} =40.21 ms what results in the minimally needed buffering of $D_{max} - D_1 = 40.21\,\text{ms} - 31.65\,\text{ms} = 8.56\,\text{ms}$.

If the first packet is delayed by $D_{max} - D_1$ before it gets forwarded, even the slowest packet will arrive just-in-time for its designated transmission ($D_1 + 8.56\,ms = 40.21\,ms = D_{max}$), while all other packets that arrive more or less before their scheduled time must be adequately buffered. Using the calculated delay, it would be possible to perform a perfectly smoothed outgoing stream. Consequently, all packets would again have the same delay characteristics as when entering the network at the first-hop router. Consequently, no distinction from a physical leased line would be possible.

Unfortunately, when the last-hop router receives the first packet of a data flow, it is unknown, which relative delay it experienced in the possible interval $[D_{min}, D_{max}]$. The last-hop-router is simply unable to perform this contemplation because there is no means to determine the corresponding delay values. This is the reason why the Virtual Wire PDB automatically assumes the worst-case, that is the first arriving packet having traversed the network with the lowest possible delay D_{min} and that it must therefore be buffered as long as necessary for the slowest packet (D_{max}) to arrive just in time. This duration $D_{max} - D_{min}$ equals exactly the maximum possible variation of the transmission delays, that is the highest possible jitter J_{max}.

In the presented example the length of a virtual packet time was 20 ms. When using the currently proposed specifiation of the Virtual Wire PDB [6] the first packet would have been buffered for this duration in the last-hop router and thereby enabling the generation of a perfectly smoothed outgoing stream (because $D_1 + 20\,ms = 51.65\,ms \geq D_{max}$). However, in this example the additional delay of 20 ms clearly exceeds the least necessary and sufficient delay of 8,56 ms (cf. fig. 6).

Based on this contemplations two alternative mechanisms for the realization of an adequate quality-of-service support for non-adaptive real-time applications will be proposed in the following and compared to the VW specification proposed by [6]:

B. Original Virtual Wire PDB

This option corresponds to the per-domain behaviour Virtual Wire as proposed in [6] and described in section II. As long

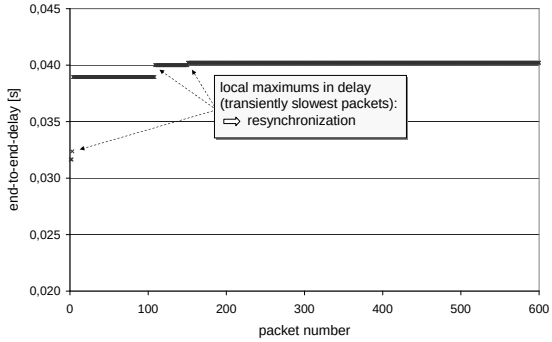

Fig. 7. Resynchronisation to a new transient local maximum in delay while shaping the data flow in the last-hop router (*adaptive shaping*)

as all existing restrictions for the deployment of the VW-PDB concerning the network configuration (topology, link capacities and parameters of all planned data flows) are met, the guarantees regarding the delay variations are surely granted.

For interactive real-time applications such a PDB is only applicable as far as the delay does not get too disturbing, which gets more improbable with smaller bandwidths (cf. fig. 3). In any case the Virtual Wire PDB in its original version is recommended for applications not able to compensate any delay variation on their own.

C. Adaptive Buffering

For applications that can temporarily tolerate small delay variations or do not need the guarantee of a continuously uninterrupted smoothed data flow, the following approach of an *adaptive buffering* is recommended. With this traffic conditioning approach the additional delay for the jitter compensation is always adjusted to the maximum observed delay of all so far regarded packets of one data flow. In this case after the arrival of packet i the last-hop router will use the following additional delay T for the remaining packets of this VW data flow:

$$T = \max_{k=1\ldots i} \{D_k\} - D_1 \tag{1}$$

Figure 7 shows the resulting end-to-end delays of all packets from the example of figure 4 experienced by the receiver when applying the concept of adaptive buffering. It is obvious that the delay of each packet that sets a new local maximum temporarily interrupts the continuous packet stream and the buffering adapts to the new maximum delay. For this purpose there is no need for complex mechanisms in the last-hop router, because this adaptation occurs almost automatically when the packet buffer runs empty and the arrival of the next packet is awaited. As soon as it arrives a new reference point for the shaping of the outgoing data stream will be set. The adaptation to the local maximums forms a transitive envelope of all occured delays what is pointed out by the arrows in figure 7.

The big advantage of this approach is the achievement of the least necessary delay for the generation of a smooth output stream. That means *at no time* an unnecessary additional delay will be added to the VW packets. Though, there are short momentarily discontinuities in the smoothness of the data flow when it adapts to a new local maximum. However the whole adaptation process is normally finished rather quickly and after a short period the generated output stream is forwarded continuously and smoothed to the receiver. In the observed example in figure 4 already after the third packet a buffering delay is reached which only differs 6.3% from the maximum delay. After 109 packets the next adaptation to a level of 98.9% from the upper delay bound takes place, which is finally reached after another 42 packets. After this phase, the receiver continuously gets a perfectly smoothed data stream, whose maximum end-to-end delay is only 40.21 ms – compared to a delay of 51.65 ms, which would be experienced when the traffic conditioning mechanisms of the original Virtual Wire PDB would be used. Additionally through the concept of adaptive buffering always the least delay necessary for the shaping of the concerned data flow is used.

Consequently, this approach can be recommended for all interactive real-time applications that can tolerate small interruptions in the smoothed transmission and that do not depend on an absolutely continous smooth data stream. Naturally this applies for the majority of interactive real-time applications. Most of them are actually able to compensate variations in the end-to-end delay on their own by buffering.

D. No buffering in the last-hop router

An even simpler modification of the Virtual Wire PDB provides *no* buffering by the last-hop router *at all*. This means, that no additional delay will be added to any data packet of a VW data stream. Instead, all arriving packets are forwarded immediately after their arrival at the last-hop-router without any buffering or traffic conditioning mechanisms. The characteristic of the data flow experienced by the receiver corresponds to the example depicted in figure 4.

Among other benefits this option has the following advantages:

- The last-hop router does not need any classification and/or configuration data. In both approaches that have been presented above the last-hop router must be able to distinguish different VW flows and know their configured data rates, which is why his complexity is similar to that of a DiffServ first-hop router.
- There are no resources necessary for the buffering, respectively traffic conditioning. In the approaches presented before, a memory space of $M_1 = (D_{max} - D_{min}) \cdot R$ for the original VW approach, respectively $M_2 = (D_{max} - D_1) \cdot R$ for the adaptive buffering approach is needed for the buffering of the packets of each single VW flow with rate R. According to the number of processed VW flows, their bandwidths and the desired jitter to be compensated, the totally needed buffer memory can increase very quickly.
- This approach adds no additional delay to the VW data packets.

The option of no buffering at all complies exactly to the guidelines of the Internet's ,,end-to-end" design principle [7], since the network is relieved from a complex mechanism – in this case the compensation of delay variations – that is moved to the corresponding end-systems. The task of jitter compensation can also be performed by the end-systems (respectively the

deployed application) instead of the last-hop router. Reasons for this approach are the following:

- The application is actually the real end-point of the communication path. As soon as data packets arrive there, they do not experience any further delays. This is important, since even on the last link from the last Diffserv node to the receiver there must be additional delays respectively delay variations taken into account, e.g. caused by packets of other network service classes that are in the state of transmission just when a VW packet shall be sent.
- In general, the end-system can provide the necessary resources for the packet buffering. If not, for example in devices with little memory capacity (e. g. mobile phones, PDAs and others) it is always possible to choose one of the alternative approaches presented before.
- The single instance that knows best about its requirements is the application itself. Therefore it can react best and most flexible to possible delay variations. In contrast to a router, the application can observe the arrival rate and adequately decrease or increase the amount of the compensation buffer space to be able to process the data as contemporary as possible. If the delay factor is critical and a minor packet loss is tolerable, it might also be an interesting option to exclude the slowest packets from further processing to reduce the latency even more.
- Virtually all applications available today that must depend on the unreliable Best-Effort service up-to-now are already employing some buffering mechanism. For example applications with playback buffers as video-streaming or voice-over-IP applications fall into this category [2]. Because of the missing guarantees of the best-effort network service, these applications could only be deployed up to a certain degree (a rule of thump might be a data rate of 100 kbps). Otherwise the delay caused by the buffering increases too much to be applicable for the used application. With the help of the guaranteed bandwidth and the small delay of the presented modifications of the Virtual Wire PDB, these applications could be used with noticeably higher data rates.

Since the user of an unbuffered VW-similar network service would have to take care about the jitter-compensation on his own, the service provider should guarantee the adherence of certain limits. E.g., it could be fixed in the contracted service level agreements that the occuring jitter and delay must meet some given (at least statistical) upper bounds.

IV. Conclusion

Table I compares the two different approaches proposed in this paper with the Virtual Wire in its original form. A network service provider has now the opportunity to deploy either only one or more of these alternatives in parallel and can offer his customers individual service guarantees. As it has been described before, each of the presented approaches has individual attributes and qualities that makes it attractive for certain applications. This points especially to the requirement of small delays, where the alternatives of an adaptive or no buffering at all obtain optimal results and the ability of an application to cope with jitter on its own. An application that is not able to handle any delay variations itself must choose the original Virtual Wire PDB. Whereas if it is possible to tolerate small skips in the equability – that by the way happens only during an initial adaptation phase – adaptive buffering would be the preferred approach. It has the advantage of an obviously smaller delay. All applications that are able to compensate jitter on their own can use the no-buffering alternative, that could probably be offered for a smaller charge due to its simplicity.

Variation of VW-PDB	delay in last-hop router: absolute delay	delay in last-hop router: unnecessary delay	shaped data flow
Virtual Wire like [6]	virtual packet-time $P_R = \frac{L}{R} = D_{max} - D_{min}$	$[0, P_R]$	✓
adaptive shaping	$\max_{k=1\ldots i}\{D_k\} - D_1$	0	✓ *(with resync)*
no shaping	0	—	—

TABLE I

Characteristics of the proposed alternative conditioning mechanisms compared to the Virtual Wire PDB

Regarding a simultaneous deployment of the three traffic conditioning alternatives it must be paid attention to the fact that the strict restrictions applying to the usage of the Virtual Wire PDB (see section I) are also valid for the alternative approaches. Otherwise it would not be possible to assure the deterministic guarantees of the delay variations. If the Virtual Wire PDB in its original form [6] is *not* used, it gets possible to circumvent these severe restrictions, since the traffic conditioning mechanisms in the last-hop router can adapt to higher jitter values. In this case it must be noted that it will only be possible to meet a statistical guarantee for a given delay variation.

References

[1] F. Baker, J. Bennet, A. Charny, B. Davie, et al. Supplemental Information for the New Definition of the EF PHB. IETF (DiffServ Working Group), June 2001.

[2] U. Black. *Voice over IP*. Prentice Hall, Aug. 1999. ISBN 0-130-22463-4.

[3] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Services. RFC 2475, Dec. 1998. IETF.

[4] A. Charny and J.-Y. L. Boudec. Delay Bounds in a Network with Aggregate Scheduling. In *Proceedings of Qofis2000 (Quality of Future Internet Services), Berlin, Germany*, Sept. 2000.

[5] B. Davie, A. Charny, J. Bennett, K. Benson, J. L. Boudec, W. Courtney, S. Davari, V. Firoiu, and D. Stiliadis. An Expedited Forwarding PHB. RFC 3246 (Proposed Standard), Mar. 2002. IETF (DiffServ Working Group).

[6] V. Jacobson, K. Nichols, and K. Poduri. The Virtual Wire Per-Domain Behavior. draft-ietf-diffserv-pdb-vw-00, July 2000. IETF (DiffServ Working Group).

[7] J. Saltzer, D. Reed, and D. Clark. End-to-end arguments in system design. *ACM Transactions in Computer Systems*, 2(4), Nov. 1984. http://www.reed.com/Papers/EndtoEnd.html.

[8] Urs Thürmann and Martina Zitterbart. IP-Telefonie über Differentiated Services. *PIK – Praxis der Informationsverarbeitung und Kommunikation*, 24(1):24–31, Mar. 2001.

[9] K. Wehrle. *Flexible und skalierbare Dienstgütemechanismen für das Internet der nächsten Generation*. Shaker, Aug. 2002. ISBN 3-832-20491-1.