On the UMTS RLC Parameters Setting and their Impact on Higher Layers Performance

Michele Rossi*, Lorenzo Scaranari, Michele Zorzi*
{mrossi, mzorzi}@ing.unife.it
l.scaranari@zuffellato.com
*DIF, University of Ferrara, via Saragat, 44100 Ferrara, Italy

***Abstract*— In this work the performance of the Radio Link Control (RLC) level of the Universal Mobile Telecommunications System (UMTS) is investigated. The role of a correct setting of link layer parameters is evaluated by considering both forward and backward channel metrics. End-to-End performance is obtained considering the TCP flow. The main findings of our investigation are that: i) a correct setting of RLC parameters is pivotal to avoid deadlock events, that would lead to a dramatic performance degradation; ii) schemes using receiver-driven retransmission requests give the best performance; iii) the EPC mechanism [1] gives very good performance in terms of delay, throughput and energy; iv) anomalous behaviors, leading to delay jitter performance degradation, are observed for some receiver-driven schemes. Again, EPC is able to overcome these problems.**

I. Introduction

The Universal Mobile Telecommunications System (UMTS) is a 3rd generation system based on the WCDMA technique, where voice, high-speed data traffic and multimedia applications can be supported. In UMTS, the Radio Link Control (RLC) layer is placed on top of the Medium Access Control (MAC); its function is to counteract the residual errors (after physical layer processing) by performing retransmissions of lost data, thus providing a reliable data transmission channel.

In this work we are interested in the characterization of higher layer protocols performance in UMTS and, in particular, in the effect of the RLC parameters setting. This understanding is pivotal for higher layer flows to fulfill the desired QoS requirements. Here, this study is carried out considering the TCP protocol as the transport layer mechanism. The TCP assumption that losses are due to network congestion becomes problematic in wireless links, where indeed losses are often due to signal degradation. The RLC layer can overcome this fact by transparently recovering from channel errors before the TCP timeout expiration occurs. However, depending on the RLC configuration, this goal can be achieved with different delay, throughput and energy consumption performance.

Several contributions on the RLC setting have been proposed so far in the literature. In [2], simulation results are reported to highlight the importance and the role of a correct setting of some RLC parameters such as the *Pool Empty TX Buffer*, *Pool Empty RETX Buffer*, *Pool Every X SDUs* and *SDU Discard Timer* [1]. In [3], instead, the influence of the maximum number of allowed retransmissions (MaxDAT) is studied in detail. The effect of the RLC window size is considered in [4] and the setting of the various RLC timers has been investigated in [5]. In this work, we go further with respect to all these contributions for the following reasons: following the guidelines derived in the previous research, we propose three different and complete RLC setting modes and we evaluate their delay, energy and throughput performance. The effect of independent and correlated losses is considered. The EPC mechanism [1] and its benefits are investigated.

The paper is structured as follows: in Section II the reference scenario is introduced, in Section III the main RLC parameters are presented and the three different RLC setting modes are specified. The performance of these RLC schemes is then evaluated in Sections IV and V, where some important metrics are shown for the independent and correlated error case, respectively. In Section VI higher layer packet delay performance is reported, by highlighting an anomalous behavior for some of the proposed schemes and finally, in Section VII some conclusions are given.

II. Reference Scenario

We consider a UMTS mobile Terminal Equipment (TE) that is communicating with its serving Base station (BS) through an uplink Dedicated CHannel (DCH). The RLC level is used between ME and RNC [6] to recover from residual errors at the output of the physical layer. Then, TE data is forwarded to a Fixed Host (FH) placed in the Internet. Errors in the RLC forward channel (where data packets flow) are modeled by means of a discrete-time two-state Markov Model that evolves frame by frame (the frame duration is 10 ms). Despite its simplicity, this model allows us to derive some precise and useful indications on the impact of error burstiness on the RLC performance. All PDUs transmitted in a bad state are considered to be lost, whereas the ones sent during good states are always correctly received. The Frame Error Rate (FER) is defined as the average probability to be in the bad state. Errors are also considered for what concerns the backward RLC flow, i.e., regarding RLC acknowledgments (called *Status Reports* [1]). Here, an

This work has been supported by ERICSSON Research.

independent error process is adopted, i.e., status messages are lost with a fixed probability, P_{status}. The RLC layer round trip time has been considered to be constant and equal to RLC RTT=100 ms. The simulation tool used to perform our performance evaluation is similar to the one described in [2], a complete description of our simulation tool is not reported here due to space constraints.

III. UMTS RLC Scheme: Configuration Parameters and Settings

In this section we first give a brief description of the RLC layer mechanism and of the main parameters involved in its configuration, then we give some guidelines for their correct setting and finally we propose three RLC configuration modes, namely, *delay*, *resource* and *throughput optimized*.

A. Basic concepts

The RLC is basically a Selective Repeat (SR) ARQ scheme where some new features have been added to improve performance and to provide a fully programmable and flexible retransmission mechanism. Packets incoming from higher layers (RLC SDUs) are segmented in RLC PDUs, that are the packet units at the RLC. A Cyclic Redundancy Check (CRC) field is attached to each RLC PDU to perform error detection. The receiver RLC entity based on the CRC field and on the gap detection technique[1] sends back to the transmitter *Status Report* messages containing a list with the PDUs requested for retransmission. The sender can force the transmission of status reports by sending special *Poll* packets[2]. At the receiver a *Poll PDU* causes the immediate transmission of a new status report[3]. This procedure is useful to avoid deadlock situation and, in general, to control the amount of acknowledgment messages sent by the receiver (backward flow). In addition, many features have been included in the RLC to reduce the retransmission delay, prevent deadlock and reduce the feedback channel overhead. Their detailed explanation can be found in the standard [1] as well as in [2][3]. The detailed explanation of the RLC parameters is out of the scope of this paper. The interested reader is referred to the references above.

B. RLC Schemes Proposal

Some general guidelines must be kept in mind when we are configuring the RLC layer:

- At least one mechanism to force the periodic transmission of status messages must be accounted for. This is necessary in order to avoid deadlock at the RLC sender. For this purpose, two mechanisms must be set: the *Poll Timer* and the *Poll Empty TX Buffer*. The *Poll Empty TX buffer* provides to the transmission of a *Poll PDU* when the last PDU in the transmission buffer is sent. Once the Poll has been sent, the *Poll Timer* forces a periodic retransmission of the Poll (when the timer expires). This retransmission process is stopped when the requested *Status PDU* is finally received. These features are set in all the schemes proposed in the next to obtain deadlock free mechanisms. Some useful insights on these settings can be found in [2].
- Poll packets should be sent frequently to reduce the retransmission delay [2]. By this way more status reports are sent back to the receiver that, as a consequence, can rely on promptly updated retransmission information. However, the feedback flow (*status reports*) is directly related to the backward channel energy expenditure. Clearly there is a trade-off of the polling frequency.
- *Sender-Driven* schemes have the advantage of controlling the sender window so as to avoid stall situations. In pure Sender-Driven approaches, it is the sender that, based on the state of the sender window, RLC queues and RLC flow at the transmitter, controls the transmission of status messages by polling the receiver.
- In *Receiver-Driven* schemes, it is the receiver that, relying on the *Check Missing PDUs* [1] mechanism, infers lost packets and sends back *status reports* accordingly, i.e., the backward flow is governed by the receiver.

We verified that *Sender-Driven* schemes are always inferior to *Receiver-Driven* mechanisms and that the best performance is achieved in the hybrid case, i.e., when both sender and receiver features are used together. In the following, referring to the concepts introduced above, three different RLC setting modes are presented:

- **Delay Optimized (DO)**: in this scheme the *status reports* transmission is controlled by the *Check Missing PDUs* feature. To avoid deadlock at the sender, the *Poll Window* mechanism must be activated, that is, the transmission of a *Poll PDU* is forced whenever the sender window increases beyond a given threshold. The aim of this scheme is to aggressively and promptly retransmit lost PDUs so as to minimize the delay. This is achieved thanks to a continuously updated (and redundant) backward status PDU flow. In particular, the receiver *Prohibit Status Timer* is set to a value lower than the RLC layer round trip time (RLC RTT). When this timer is activated, *status PDUs* can be periodically sent at its expiration times. Setting the timer value is a method for controlling the maximum number of *status messages* flowing back in a RLC RTT. Finally, in the DO scheme, a *Poll PDU* is sent when the retransmission buffer is empty (*Poll Empty RETX Buffer* feature).
- **Resource Optimized (RO)**: this scheme is similar to the previous one except for the fact that the *Poll Empty RETX Buffer* is not set and the *Prohibit Status Timer* is set to a value greater than RLC RTT. In this way at most one *status*

[1] The received PDU flow is checked for discontinuities in the sequence numbers.

[2] A Poll bit is set in the normal data PDU.

[3] Unless the *Prohibit Status Timer*, whose function is explained in the following, is set.

SCHEMES →	RO	DO	EPC
RLC transmitter params			
Poll Empty TX Buffer	yes	yes	yes
Poll Timer	120 ms	120 ms	150 ms
Poll Every X PDUs	no	no	no
Poll Every X SDUs	no	no	no
Poll Empty RETX Buffer	yes	no	no
Poll Window (%)	80 %	80 %	80 %
Periodic Poll	no	no	0
Prohibit Poll Timer	no	no	no
RLC receiver params			
EPC	no	no	yes
Prohibit Status Timer	60 ms	110 ms	110 ms
Check Missing PDUs	yes	yes	yes

TABLE I
RLC SCHEMES CONFIGURATION PARAMETERS.

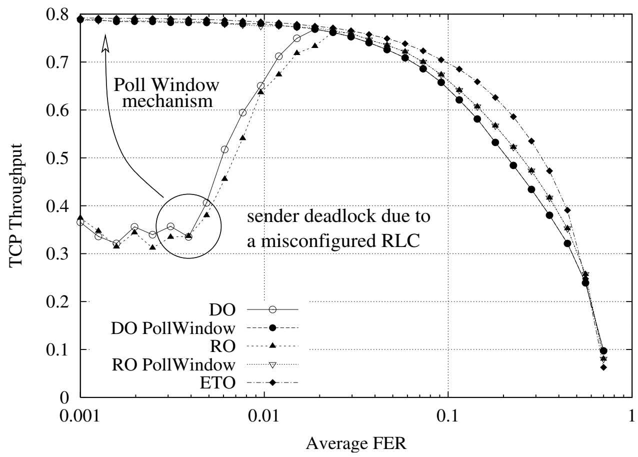

Fig. 1. TCP throughput comparison between the selected schemes in an independent channel, 100 Kbyte file transfer.

message can be sent in every round trip time. This scheme has the advantage of saving bandwidth (and energy) in the feedback channel, but this comes at the cost of a lower retransmission promptness and, as a consequence, of longer RLC SDU delays.

- **EPC throughput optimized**: both the EPC mechanism [1] and the *Check Missing PDUs* are enabled. The *Poll Window* mechanism is triggered to send a *Poll PDU* whenever the sender window reaches 80 % of its maximum size.

These settings are summarized in Table I.

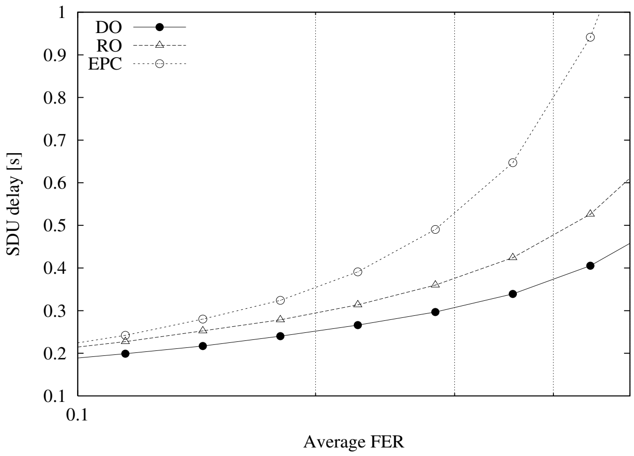

Fig. 2. SDU delay in an independent channel, 100 Kbyte file transfer.

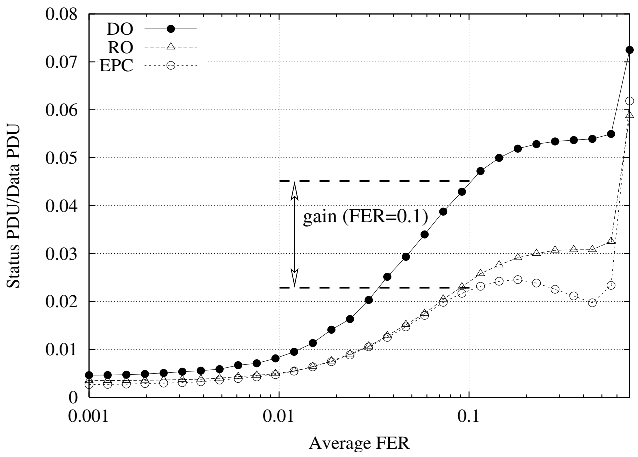

Fig. 3. Feedback channel efficiency, 100 Kbyte data transfer.

IV. COMPARISON BETWEEN RLC SCHEMES IN AN INDEPENDENT CHANNEL

In the results reported here, we focus on FTP data transfer performance, i.e., we consider a continuous data source modulated by the TCP window mechanism. Two data transfers, of 5 and 100 Kbyte are taken into account to highlight the role of the TCP Slow-Start phase. The discontinuous transmission phase, due to TCP, is dominant in the 5 Kbyte case, where the Slow-Start is dominant, whereas in the 100 Kbyte case, the discontinuous transmission is less influential since here the TCP window size is large enough to fill the transmission channel. We have performed intensive simulations to characterize the proposed schemes. In the sequel, due to space constraints, we only report some of them to highlight the main findings of our investigation.

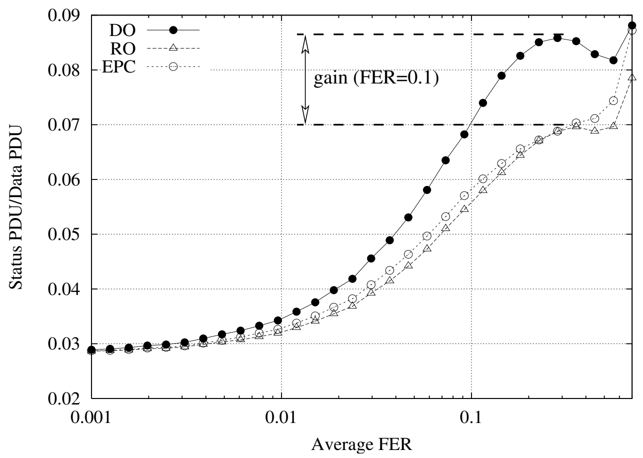

Fig. 4. Feedback channel efficiency, 5 Kbyte file transfer.

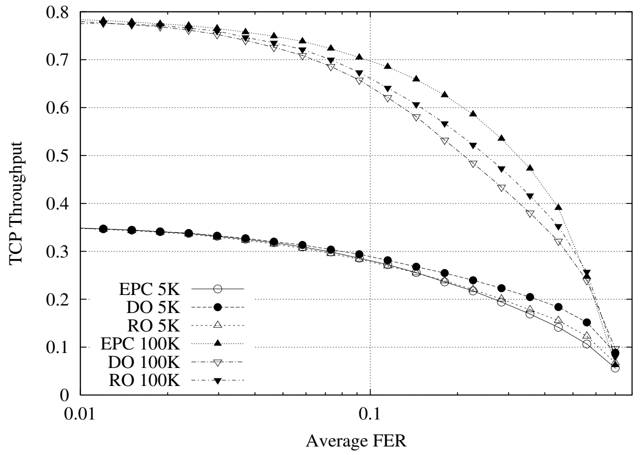

Fig. 5. TCP throughput as a function of the FER. 5 and 100 Kbyte FTP data transfers.

In Fig.1 we compare the throughput for the schemes presented in Section III-B. To show the importance of the window control feature, the RO and DO schemes are also reported here in their version without window control (i.e., where the *Poll Window* mechanism is not activated). It appears evident how the throughput in that case is heavily degraded at low FER. The reason for this phenomenon is that the transmission of *status PDUs* is triggered by the detection, at the receiver, of missing PDUs, but when the error probability is low this is a rare event. As a consequence, few status reports are sent back to the transmitter and the RLC window may be saturated[4]. The mean SDU delay (100 Kbyte case) is reported in Fig. 2. From that figure, it is clear how the DO scheme can effectively reduce the SDU delivery delay. The EPC scheme for this metric is the one with the worst performance, whereas the RO performs in the middle. The same relationship among schemes is maintained for the 5 Kbyte case (not shown here). In Figs. 3 (100 Kbyte) and 4 (5 Kbyte) we focus on the channel efficiency defined as the ratio between the number of transmitted data PDUs (forward flow) and the number of status PDUs flowing over the backward channel, i.e., the average number of status messages sent back to the transmitter for every transmitted data PDU. It is worth noting that the EPC scheme is able to save bandwidth in the feedback channel; in both the 100 and 5 Kbyte data transfer cases DO is the scheme with the highest feedback channel utilization. Referring to $FER = 0.1$, the relative saving obtained by RO and EPC schemes is roughly 17 % and 50 % for the 5 and the 100 Kbyte transfer case, respectively. In Fig. 5, we report the TCP throughput for the 100 and the 5 Kbyte transfer case. From the results presented in this section the following conclusions can be drawn:

[4]The role of a status message is also to advance the transmission window by allowing the transmitter to release from the buffer those packets that have been correctly received.

- For what concerns the TCP throughput metric, EPC is the best scheme for the 100 Kbyte case, i.e., for large file transfers where the continuous transmission dominates, whereas DO performs better for short file transfers. The reason behind this is that when the Slow-Start phase dominates, i.e., when the transmission is discontinuous, it is important to decrease the feedback delay in order to permit a fast TCP window increase. When the TCP window size is large enough to fill the channel this behavior is no longer observed. In this last case, instead, a lower delay comes with a lower throughput as well.
- The RO scheme for all the considered metrics presents an intermediate behavior.
- The EPC scheme is the one where feedback messages are sent back with the highest efficiency, for both short and large file transfers. This scheme presents the highest throughput in the 100 Kbyte case, whereas its throughput is the lowest for short file transfers. The reason behind this is that optimizing the number of status reports flowing back to the transmitter leads to slightly longer delays in the TCP window advancement phase during the Slow-Start discontinuous phase.

V. Comparison between RLC Schemes in a Correlated Channel

In this section we report some results on the effect of the channel correlation on throughput and SDU error probability performance for the schemes introduced above. We utilize a Two State Markov Channel model to characterize the UMTS Frame error sequence, as explained in Section II. An interesting result is reported in Fig. 6, where the FER is fixed at a high value (FER=0.5) and the burst length (b) is varied in the set $b \in [1/(1 - FER) = 2, 30]$. From that figure, we can observe that the throughput curve increases for low b values, whereas it starts decreasing as b increases. This is due to two opposite behaviors. On the one hand, an increased b leads to a lower SDU delay. On the other hand, an increased b always leads to an increased SDU error probability[5]. The shorter SDU delay leads to a faster Slow-Start phase, and to the positive slope of the throughput curve in Fig. 6, for low b values. This effect is more notable for short file transfers, where the Slow-Start phase dominates. Moreover, as b increases the probability that a SDU is discarded increases as well. A SDU is discarded when one of the PDUs composing it reaches the maximum number of allowed retransmissions without having been correctly delivered. The increased SDU error probability leads to a reduction of the throughput, and this effect dominates in the second region of the curve in Fig. 6, i.e., for large b values. In Fig. 7, we report the SDU error probability as a function of the burst length. It is worth noting here that the EPC scheme is almost insensitive to an increased burst length. The explanation of this fact is found in the way this mechanism works [1]. In more detail, a timer is used to account for the time elapsed between the transmission of a *status report* and the reception of the first retransmitted PDU. This timer is usually set to an estimate of the RLC RTT. When this timer expires and no PDUs are received as a reply to the previous status report, then a new status report is transmitted and the timer is recomputed based on a new RTT estimate. In practice the number of status reports sent and their sending times is adaptively set based on the number of successfully retransmitted PDUs and on the estimated time between the sending of the status report to require their retransmission and their correct reception[6]. In the presence of a large burst of errors, the EPC at each RTT adaptively increases the inter-status sending time period, so that fewer status PDUs are sent with respect to the DO and RO schemes.

From the results presented in this section we can conclude that the EPC timer mechanism is able to manage the *status report* flow in a better way. The decision on the *status report* sending times is based on the detection of erroneous/lost PDUs (by means of the *Check Missing PDUs* feature) as well as on a consistent estimate of the RLC RTT[7]. This leads to a more efficient utilization of the feedback channel that in turn leads to:

- Increased throughput for large file transfers.
- Higher robustness against error bursts.
- Higher stability in terms of throughput performance under correlated errors.

VI. Jitter Performance

In this section we consider the delay jitter metric and we show how the EPC mechanism is able to obtain superior performance also in this case. The jitter is defined here as the mean delay between the release (at the RLC receiver entity) of SDU packets. In Fig. 8 we report the delay jitter standard deviation as a function of the FER considering a *status PDU* error probability (feedback channel packet error probability) of $P_{status} = 0.05$. In the same figure, the mean jitter delay is reported for comparison. These results have been obtained by means of an accurate UMTS channel characterization. A UMTS system simulator has been developed [7] accounting for power control, propagation phenomena (path loss, shadowing and fading), user interference and mobility. The results in Fig. 8 are relative to a scenario where $N_u = 100$ users are randomly placed in a coverage area of 25 UMTS cells (wrapped around to avoid border effects) considering a slow fading condition, i.e., every user is characterized by a Doppler Frequency of 6 Hz.

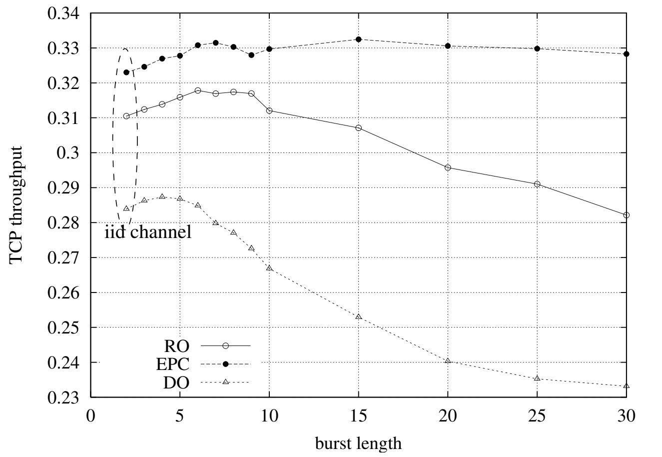

Fig. 6. TCP throughput as a function of the error burst length, 100 Kbyte FTP.

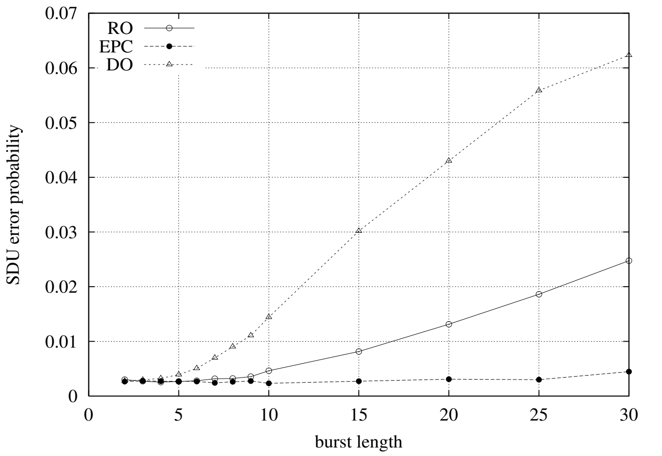

Fig. 7. SDU error probability as a function of the burst length, 100 Kbyte FTP.

[5] In general, especially for high FER this metric is greater than zero because the maximum number of retransmissions per PDU (MaxDAT) is set to a finite value. In this paper we consider MaxDAT=10. The effect of the MaxDAT parameter has been studied in detail in [3].

[6] For more details on the EPC mechanism the reader is referred to [1].

[7] Intended here as the time elapsed between the transmission of a status report and the reception of the first retransmission elicited by that status report.

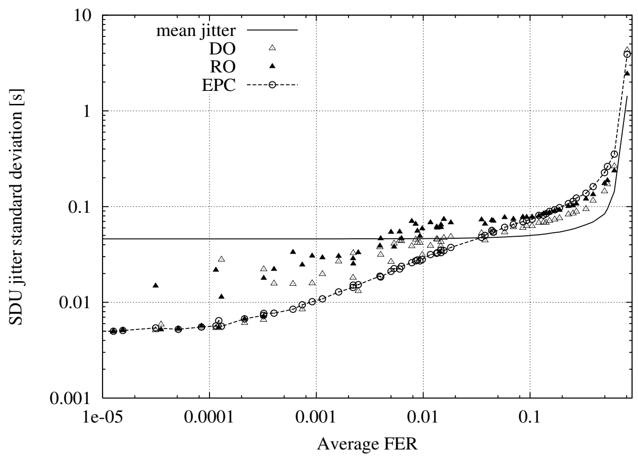

Fig. 8. SDU delay jitter standard deviation as a function of the FER.

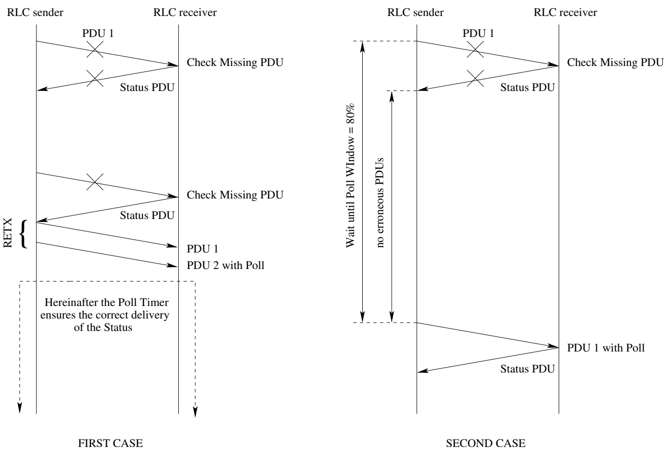

Fig. 9. Error recovery in DO and RO schemes.

Looking at Fig. 8 an anomalous behavior can be observed for the DO and the RO schemes. At low FER, their standard deviation is very high and as large as the mean jitter value. This behavior can be explained using the example situation in Fig. 9. In the first case (Fig. 9) we consider that two PDUs in a row are lost. The status report relative to the first PDU is lost, whereas that relative to the second packet is correctly received. Upon reception of the last status report, the two erroneous PDUs are retransmitted and a Poll is inserted in the second PDU (thanks to the *Poll on Empty RETX Buffer* feature). In the second case, only one PDU is lost and its status message is lost as well. At this point, no mechanism exists to ensure the correct reception of the status report. Suppose that no further PDUs are corrupted. Then, a *Poll PDU* is forced when the sender window reaches the *Poll Window %* of its maximum value. And only this *Poll* can trigger a new status report to be sent. In this case, a long time is needed to recover from the error since no mechanism is adopted to ensure the correct reception of the first status report. This causes the sender window to advance, by transmitting new packets and without performing the retransmissions requested in the lost *status*.

Once again, the EPC is able to overcome this problem. In fact, due to the EPC algorithm, reports are automatically resent based on round trip time estimates. The receiver, in this case, does not have to wait for the reception of a correct *Poll PDU* to re-schedule the *status message* retransmission.

VII. Conclusions

In this paper some guidelines for the setting of the UMTS RLC parameters have been presented. Moreover, three different schemes have been proposed so as to minimize the delivery delay (DO scheme), the throughput (EPC scheme) and to achieve a trade-off between these two metrics (RO scheme). From the obtained results it is clear how the RLC setting can highly impact the higher layer performance. Based on the delay and throughput metrics, we suggest to use delay optimized schemes (DO) when the transmission is mainly in the discontinuous phase. In this case, the solution that minimizes the delivery delay also minimizes the throughput, when window based protocols (such as TCP) are considered at the transport layer. Regarding the energy consumption, the EPC scheme always represents the best choice, whereas DO is the worst. The RO scheme performs in the middle for all the considered metrics. Based on these observations a new possibility opens up, that is to use delay optimizing schemes when the transmission is discontinuous and to switch to energy optimized solutions when the channel is filled by transmitted packets. As a last conclusion, we observe that for jitter delay sensitive applications, the EPC scheme, among the schemes proposed in this paper, is the only one with a stable behavior.

References

[1] 3GPP TS 25.322, "Third Generation Partnership Project: Technical Specification Group Radio Access Network; Radio Link Control (RLC) Specification (Release 5)."

[2] Su Hsuan-Jung, Q. Zhang, "Performance of UMTS Radio Link Control," Proceedings of IEEE ICC, Vol. 5, pp. 3346-3350, April 2002.

[3] F. Lefevre, G. Vivier, "Optimizing UMTS Link Layer Parameters for a TCP Connection," IEEE Vehicular Technology Conference, Vol 4. pp. 2318-2322, May. 2001.

[4] X. Hua, C. Yi-Chiun, X. Xiao, E. Gonen, L. Peijuan, "Performance analysis on the radio link control protocol of UMTS," IEEE Vehicular Technology Conference, Vol 4. pp. 1755-1759, Sept. 2002.

[5] X. Xiao, C. Yi-Chiun, X. Hua, E. Gonen, L. Peijuan, "Simulation analysis of RLC timers in UMTS systems," Proceedings of the Winter Simulation Conference, Vol. 1, pp. 506-512, Dec. 2002.

[6] Third Generation Partnership Project Web-site: `http://www.3gpp.org`

[7] A. Giovanardi, G. Mazzini, V. Tralli, M. Zorzi, "On the effect of Doppler bandwidth, network load and power control threshold on the UMTS/FDD radio interface performance," Proceedings of IEEE PIMRC, Vol. 2, pp. 18-21, Sept. 2000.