TSG-RAN Working Group 2 (Radio layer 2 and Radio layer 3) ***TSGR2#4(99)395***

Berlin, 25th to 28th May 1999

Agenda Item:

Source: Philips

Title: Suspend & Resume to support hand over, channel type switching (updated)

Document for: Decision

---

1. Introduction

On the TSG-RAN-WG2 meeting #2 in Stockholm, Nokia presented a method for RLC-PDU Header Compression [1], which is grounded on semi-static-size payload units as the basic segmentation unit of the RLC re-transmission protocol and the possibility to include a variable number of payload units to an RLC-PDU.

Despite doubts by Ericsson, Philips, and Nortel concerning the efficiency of this scheme, the meeting minutes of TSG-RAN-WG2 meeting #2 say that this scheme was accepted as mandatory for the UE, while its use had to be negotiated at RAB setup.

Obviously, the scheme was adopted without fully understanding its pros and cons. This contribution analyses the scheme further, and proposes the suspend & resume scheme known from GSM [3] as a basic means to cope with channel type switching, and bit rate change in case of hand over or without hand over.

2. Recap of the concept of semi-static-size payload units with header compression

In the concept [1] an RLC PDU, that is transmitted via a logical channel to the receiving RLC entity, is built up from a number of payload units (PU). A PU size is determined by the lowest data rate available on the transport channel, onto which the logical channel is mapped: in case the RLC-PDU only consists of one PU, the PU plus the header must fit into the smallest transport block of the underlying transport channel. As an example for the PU, [1] mentions 10 octets.
If the RLC-PDU consists e.g. of 4 PUs in sequence, and there is a transport format defined such that a PDU may contain 4 PUs, the 4 PUs plus necessary header must fit into one transport block of the corresponding transport channel. If the PUs are in sequence, the necessary header can be compressed with the technique described in [1].
If the RLC-PDU consists of PUs, which are not in sequence, for each of the PUs an extended header has to be used. Due to this extended header, the resulting RLC-PDU contains (at least) *one PU less* than in case of sequential PUs with header compression applied, and the overhead is considerably increased.

The question arises, when the extended header is used. A closer look at the concept shows the following (For the general data flow see Fig. 1.):

1. Case: The radio channel conditions are nice, and no re-transmissions are necessary.
   ⇒ The compressed header can be used.

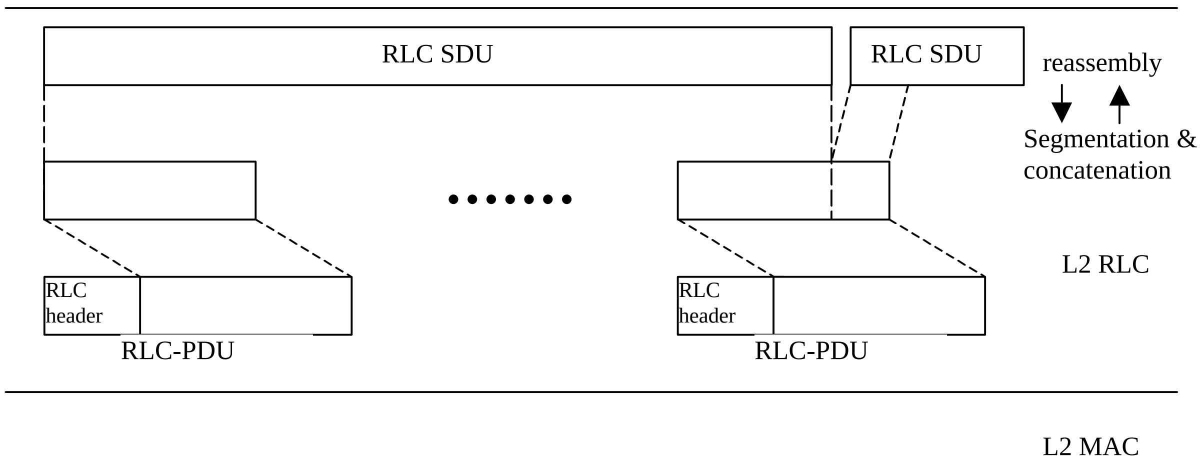

Fig. 1 General data flow for RLC SDUs according to S2.01. Whether PUs are used or not is not depicted here.

2. Case: The radio channel conditions are poor, re-transmissions are necessary, and the number of PUs to be sent in one RLC-PDU does not change (i.e. no channel type switching or change in channel bit rate.)
   ⇒ The compressed header can be used, because if e.g. out of 10 RLC-PDUs containing 4 PUs, every second RLC-PDU has to be retransmitted, e.g. PU No. 5,6,7,8 and PU No. 13,14,15,16 have to be retransmitted, the PUs to be resent within the same RLC-PDU are still in sequence, although the whole set of PUs to be resent in several RLC-PDUs are not in sequence.

3. Case: The radio channel conditions are poor, there are re-transmissions necessary, and the number of PUs to be sent in one RLC-PDU changes *during re-transmission,* where not all RLC-PDUs in sequence have to be retransmitted.

   a) Number of PUs per RLC-PDU is reduced (lower available bit rate).
   PUs which originally fitted into one RLC-PDU have now to be put in more than one RLC-PDU. If e.g. PU No 4,5,6,7 and PU No 12,13,14,15 need to be retransmitted now in RLC-PDUs that carry only 3 PUs, the PUs 7, 12,13 will be put into one RLC-PDU and therefore are out of sequence. If the RLC-PDU with reduced length carries only 2 PUs, the PUs are still all in sequence within one RLC-PDU.

   b) Number of PUs per RLC-PDU is increased (higher available bit rate) .
   If the PUs meintioned in a) can be transmitted within an RLC-PDU of 6 PUs, PUs 4,5,6,7,12,13 will fit into one RLC-PDU, and are therefore out of sequence. If the RLC-PDU size is increased even to 8 PUs, PUs 4,5,6,7,12,13,14,15 fit into one RLC-PDU and are also out of sequence.

   ⇒ If the number of PUs, which can be accommodated within one RLC-PDU, increases, the extended header must be used in some RLC-PDUs.
   If the number of PUs, which can be accommodated within one RLC-PDU decreases, the extended header must be used in some RLC-PDUs, if the reduced number of PUs is not an integer divisor of the original number of PUs.

Result: The extended header will only be necessary, if the number of PUs per RLC-PDU (i.e. the transmission rate) changes during retransmission.

According to the above discussion, the efficiency of the scheme seems to be good, if the extended header will occur only rarely. In other words, if the rate changes are rare, the overhead reduction by using the compression technique is only marginal. This implies on the other hand, that if it is claimed that the extended header only very seldom occurs, the rate changes cannot be very often so that the efficiency gain becomes questionable.

In addition to the doubtful efficiency, the very high implementation complexity with segmentation and partial reassembly in the RLC layer of the sending side has to be kept in mind.

3. Using fixed-size RLC-PDUs and suspend/resume functionality

Using fixed-size RLC-PDUs, the implementation complexity is dramatically reduced. In this case, the RLC-PDU length can e.g. be determined by the maximum possible block size in the TFS of the transport channel, which is used for transmitting. In order to reduce zero padding (section 2.3 in [1]), the RLC-PDU could also be chosen taking the possible block sizes and the length of the RLC-SDU into account, and using the very block size in the TFS, which results in the lowest zero padding.

[1] states further cases, in which the concept of *semi-static-size PUs with header compression* is claimed to be of advantage:

1. Switching between dedicated and common transport channels
2. Limitations in DCH usage
    a) a dedicated channel for packet transmission in the same cell might not be usable on a higher bit rate for its whole life-time so that intermediate bit rate changes might be necessary.

    b) in case of inter-RNC hand over, the target RNC cannot support the high bit rate that was available in the original cell

All these cases can be coped with also by using *fixed-size RLC-PDUs together with the suspend & resume functionality* (if the UE is sending) efficiently. In case the UTRAN is sending no suspend & resume is needed, because UTRAN can control itself.
The description of the RLC protocol [2] already contains the suspend & resume functionality as an FFS item. Suspend & Resume (in case of the UE being sending) would already provide the functionality to allow channel type switching as well as changing the bit rate in case of hand over or without it:

As soon as there is a need for changing the RLC-PDU (and bit rate) the UTRAN would send to the UE a suspend directive, which makes the UE stop segmenting and then lets the UE only exhaust its necessary re-transmissions – using the original RLC-PDU size - to finish the transmission of the original RLC-SDU, that is not yet fully received by the UTRAN-RLC. After that the UE would reconfigure its RLC entity in order to transmit with changed RLC-PDU size, which is optimized for the transport block size of the new transport channel bit rate. On the resume signal from the UTRAN, the UE would again start segmenting and transmit with the changed RLC-PDU size (and bit rate).
In case of an inter-RNC hand over, a new RLC entity has to be created in the new RNC, i.e. an initialization procedure between the UE-RLC and the UTRAN-RLC would be necessary in addition, as it is already the case in GSM.

The differences between using the semi-static-size PUs with header compression (Nokia solution) and fixed-size RLC-PDUs with suspend & resume (Philips solution) are the following:

a) In the Nokia solution, the bit rate (and RLC-PDU size) can be changed before the original RLC-SDU is completely received at receiving side, while in the Philips solution, the bit rate will be the same until all RLC-PDUs of the original RLC-SDU have been received correctly.

b) The sending RLC protocol machine in the Philips solution has to determine a new transport block size for the new bit rate, which might take a little while. There is no need to convey the changed block size to the receiving RLC entity, because this parameter is not used for reassembly.
On the other hand, the Nokia solution will need some time to compute the extended header if necessary.

c) The amount of zero padding for the last RLC-PDU in the Philips solution might be a bit higher than that of the Nokia solution.

Let us now assess the differences:

Ad a) Though in the Nokia solution it is possible to directly change to the new bit rate by simply adjusting the number of PUs to be put in an RLC-PDU, this *does not mean* on the other hand that the Philips solution will always be slower than the Nokia solution: From application point-of-view, the original RLC-SDU on the receiving side can only be processed, *if all RLC-PDUs have been received* so that the original RLC-SDU can again be reassembled to be given to the higher layers. Here, the Philips solution is even faster, if the bit rate is to be changed from a high one to a lower one, because the RLC-PDUs are sent with the higher rate until the RLC-SDU on the receiving side is completed.
Furthermore, packet transmission is normally not delay critical.

Ad b) Since the RLC protocol machine will most probably be implemented in HW in order to be as fast as possible, this additional delay of the Philips solution should be marginal. The same seems to hold for the additional delay of the Nokia solution due to computation of the extended header if needed.

Ad c) The amount of zero padding is difficult to compare, because it would have to be compared with the extra overhead due to the extended headers, which is quite difficult to state about.

4. Conclusion

The Nokia proposal of the concept of *semi-static-size PUs with header compression* [1] was analysed with respect to its efficiency and complexity.
An alternative scheme using fixed-size *RLC-PDUs and suspend & resume functionality* was described. From the comparison it turns out that the Nokia proposal though looking really smart, is *not really superior* to this alternative, but shows a much higher implementation complexity. For hand-over cases, in which the hand-over is not seamless but hard (street corner effect), the suspend & resume functionality is needed anyway.

To get a complete picture, it has to be asked, how frequent there would be switching between channel types, as well as changing the bit rate during transmission, and how often there would be a hand over. We think that both cases are relatively rare so that the additional complexity of the Nokia proposal does not justify to make it mandatory for the UE in the sense that all UEs have to implement this scheme.

Furthermore, the fact that the header compression technique is currently assumed to be mandatory poses some strange implications: Since every UE has to implement the scheme, this would also hold for a simple voice phone, just because the UE from time-to-time uses AM perhaps on some very low rate control channel. Would this justify the increased complexity of the scheme to be posed upon such a simple terminal ?

There might be some benefit for applying the scheme for UE generating mainly packet traffic. But if the compression technique were mandatory for a class of UEs, how to specify, when a UE is “packet-mode-enough” . UMTS is a very versatile system allowing hundreds of types of terminals from very simple ones (even below speech: pager only ?) to very high-end ones, e.g. laptops with life-video-camera running several connections at the same time.

5. Proposal

Since the additional complexity of the Nokia concept of *semi-static-size PUs with header compression* is not justified in view of its relatively marginal benefits, the following is proposed:

1. Move suspend & resume to the list of items in S2.22, which are *no longer* FFS,
2. allow the header compression technique of [1] as a possible scheme, which can be used by advanced types of UE, but need not be implemented by all UEs.
3. adopt suspend & resume as a basic means to implement channel type switching and bit rate change with and without inter-RNC hand over.

6. References

[1] TSGR2#2(99)115, “Method for RLC-PDU Header Compression”, Nokia

[2] TS RAN S2.22 V0.0.2 (1999-03), “Description of the RLC protocol”, editor CSELT.

[3] ETSI TC-SMG, “GSM 04.06 Data Link layer specification”, September 1994.