

Source: Nokia

## Packet Traffic Characteristics in GPRS

### 1. INTRODUCTION

GPRS RLC/MAC procedures were designed for nonreal-time data transfer where the data arrives as one large block. However, the true nature of packet traffic is usually different from this assumption. Application protocols (e.g. SMTP, IMAP) transfer signaling messages between client and server. On the other hand transport layer TCP peer entities need to setup TCP connection and acknowledge the sent data packets. GPRS itself generates different kinds of signaling messages. This kind of behavior leads to high number of TBF establishments and releases. This causes delays in data transfer and increases load on (P)CCCH channel.

The purpose of this document is to illustrate the problem by showing some typical message flows and test results. In a separate work item proposal document a new work item is proposed to solve the identified problem in release 4 standard.

### 2. TCP CONNECTION ESTABLISHMENT

TCP connection is started with a so-called three-way handshake. First the requesting end sends a SYN segment specifying the port number of server that the client wants to connect to and the client's initial sequence number. The server responds with its own SYN segment containing the server's initial sequence number. The server also acknowledges the received SYN. Finally, the client acknowledges server's acknowledge SYN segment. This completes the connection establishment.

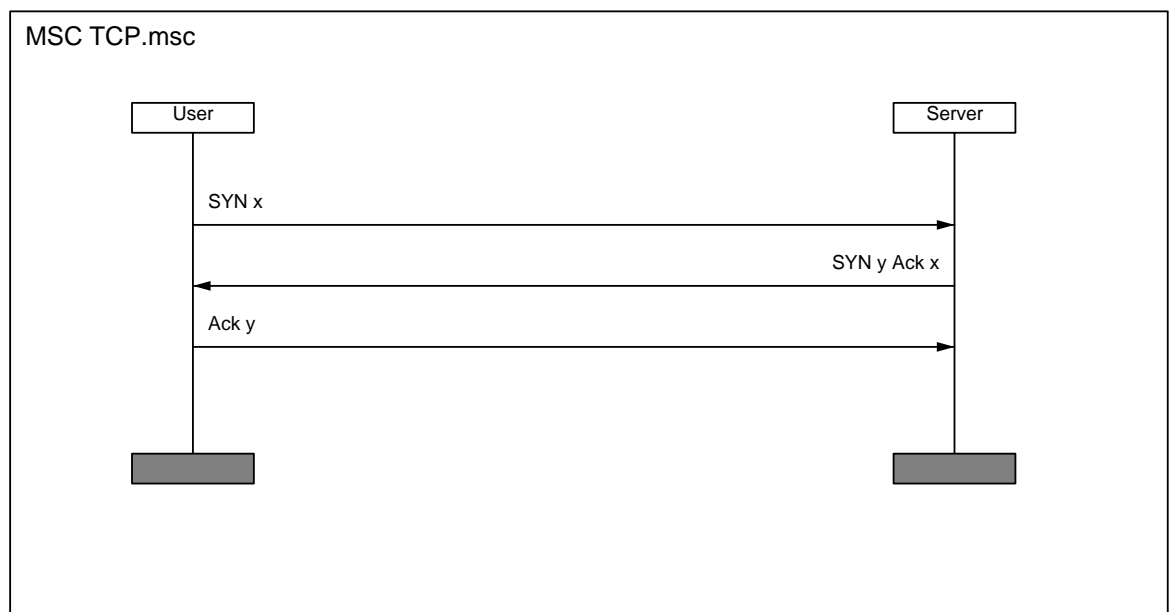


Figure 1. TCP handshake

The three segments of the TCP connection establishment are each transmitted over the GPRS radio interface in different TBFs. This can be seen in figure 2 where RLC/MAC level

message exchange has been illustrated. In this figure LLC unacknowledged mode has been assumed to make the diagram simple and illustrative.

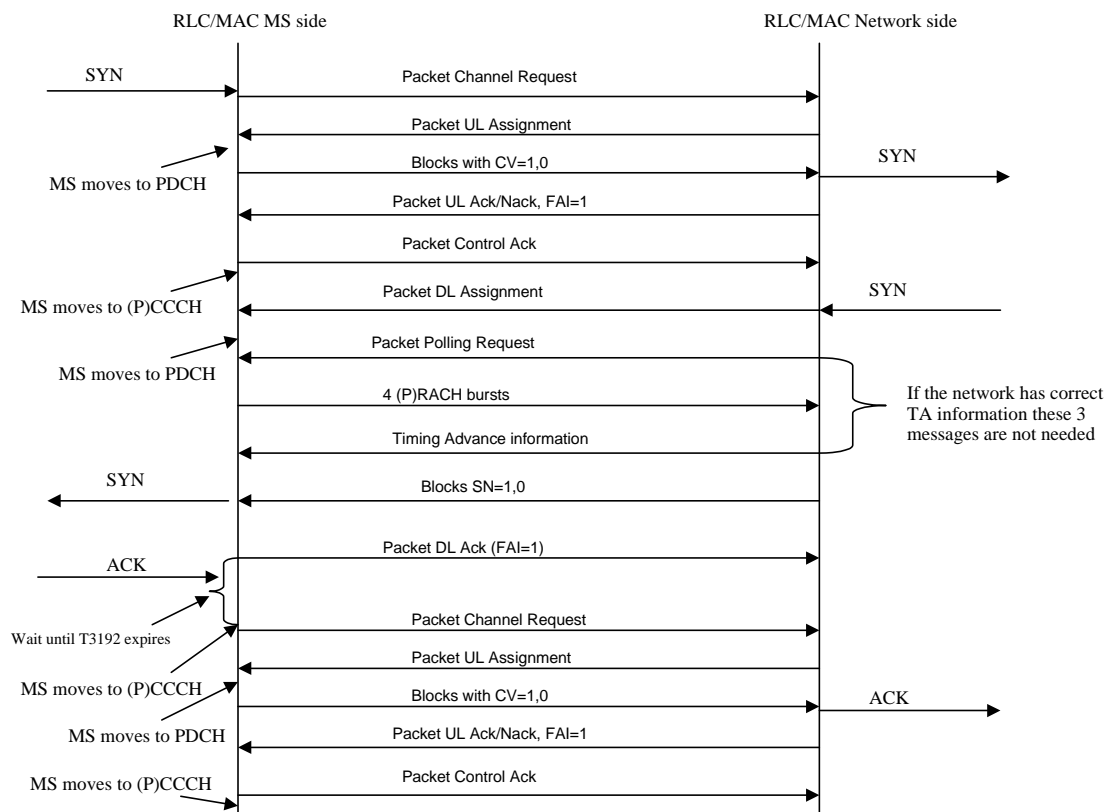


Figure 2. TCP three-way handshake on RLC/MAC level (LLC unack mode assumed)

It should be noticed that the exact message flow depends on the timing of the messages (e.g. whether TBF can be setup using PACCH or (P)CCCH) and different implementation related issues. In the figure it has been assumed that the network does not store timing advance of the MS after it has released the TBF. It has been also assumed that the last TCP ack message does not arrive before the Packet DL Ack (FAI=1) has been sent. If it arrives before the ack is sent the UL TBF can be setup by using DL PACCH.

Figure 2 shows that already the three messages of the TCP connection setup generate a lot of signaling in the RLC/MAC layer. Connection setup is delayed significantly by the TBF setups. In addition between the reception of the second SYN and the transmission of the last SYN(ack) message the MS must wait<sup>1</sup> until T3192 expires before it may move to (P)CCCH and send the Packet Resource Request.

### 3. E-MAIL

TCP handshake is only part of the session setup before the actual data transfer may begin. Also client and server need to communicate with each other. Figures 3 and 4 present examples of message exchange flows of IMAP and SMTP protocols that are commonly used for sending and reading of email from mail server. Another example has been presented in Appendix A where a trace from an e-mail session has been presented. Mail server was Microsoft exchange server and the client Netscape web browser which used IMAP protocol for downloading e-mails. The trace was taken between Netscape client and TCP layer. The results should be interpreted so that the text that follows word "client" is a message from client that is given to TCP layer for transport<sup>2</sup> and the text that follows

<sup>1</sup> If the ack arrives before the RLC DL Ack is sent then the MS need not move to PCCCH but it can set up the uplink TBF by using PACCH.

<sup>2</sup> In some cases TCP may combine several messages into one larger TCP packet.

"server" is what netscape client receives from the TCP layer (TCP layer characteristics are not visible in this trace). Interactive client-server message (IMAP) exchange can be clearly noticed. Due to the interactive nature of these messages similar behavior as what is seen in figure 2 can be expected throughout this message flow i.e. TBFs are setup and released for each of the messages and after releasing the downlink TBF the MS must wait<sup>3</sup> until T3192 expires before moving to (P)CCCH to carry out Packet Channel Request.

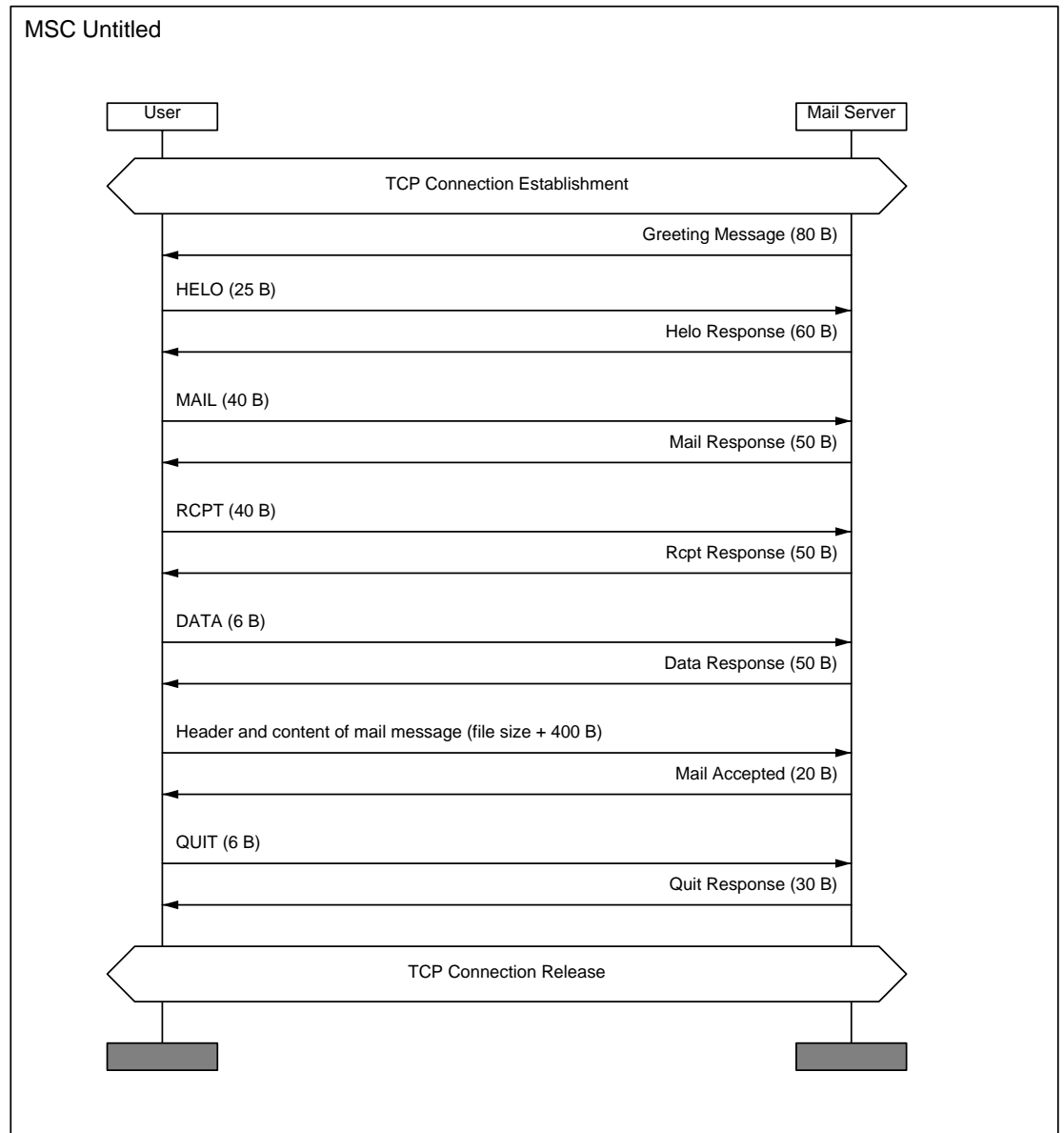


Figure 3. SMTP Signaling flow

<sup>3</sup> If the response arrives very fast the uplink TBF can be setup using PACCH.

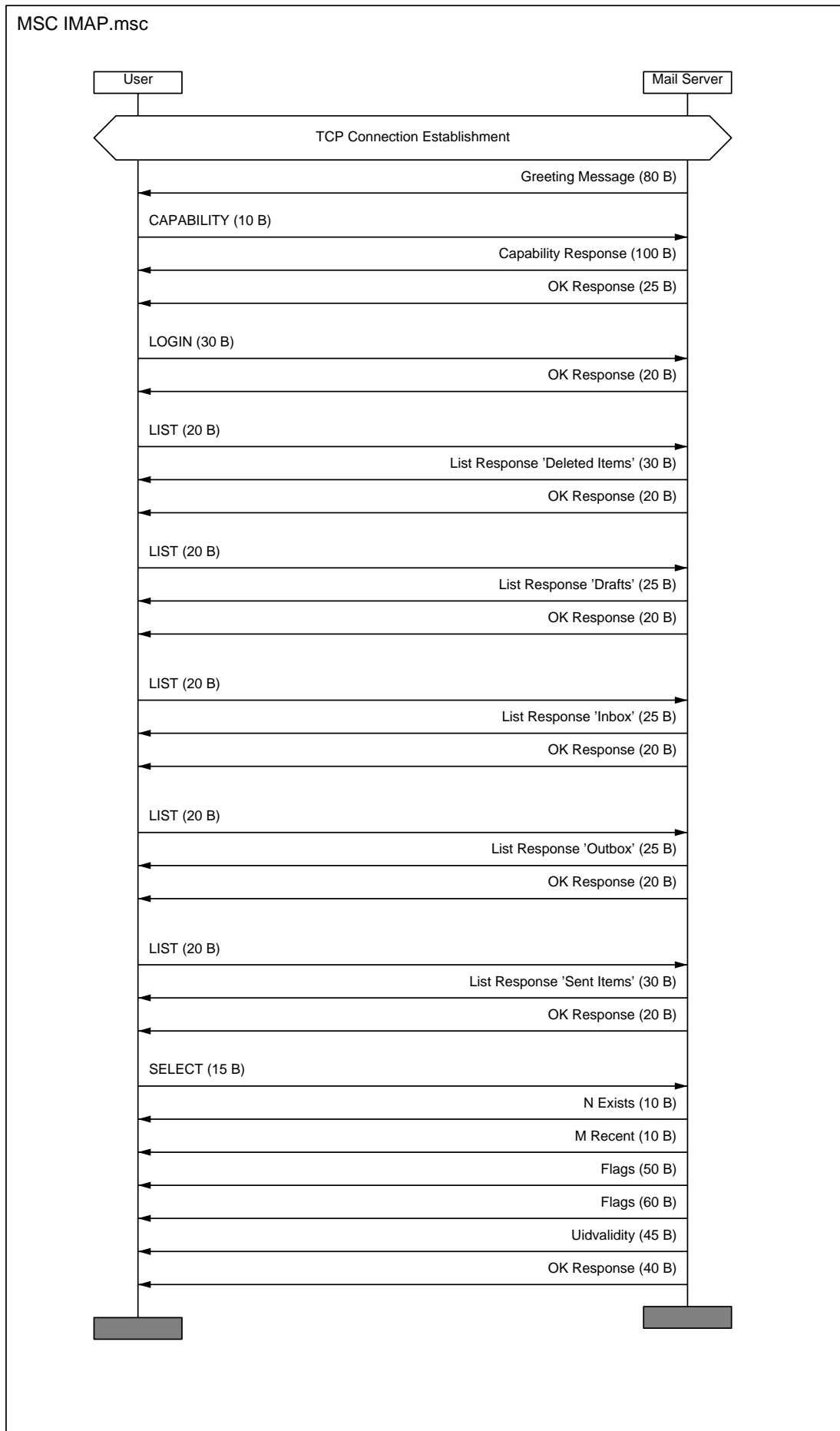


Figure 4. IMAP Signaling flow

## 4. TEST RESULTS

Some tests were carried out in workstation environment assuming GPRS protocol stack. Different applications were run over GPRS protocols in a way that the environment from the application point of view corresponds to a real GPRS network. Perfect transmission over radio was assumed using following assumptions: 3+3 GPRS time slot configuration, CS-2, countdown value 15, TBF release timer 5s, DL (P)CCCH delay 320+/-5ms.

### 4.1 Test cases

#### 4.1.1 E-mail

Used mail program was PINE (IMAP protocol) and the mailbox located in Microsoft Exchange server. The following operations were performed:

0. Contact to the mail server (user name, password, etc.).
1. Compose a short message (ten line text) without attachment and send it to itself.
2. Compose message with 46 Kbytes attachment and send it to itself.
3. Compose message with 100 Kbytes attachment and send it to itself.
4. Open inbox.
5. Read the received message (sent in 1).
6. Read the received message (sent in 2).
7. Save the attachment received in 6.
8. Read the received message (sent in 3).
9. Save the attachment received in 8.
10. Close inbox (mail were deleted).

#### 4.1.2 NetMeeting

The test was performed using NetMeeting in two machines and playing CDs in both of them. One machine was mobile station and another in wired network. CD playing time was two minutes.

#### 4.1.3 Real Player

Data stream was encoded at 20 kbit/s. Playing time was two minutes.

### 4.2 Results

Table 1 presents how many (P)CCCH messages were sent in each test for both normal TBF release and delayed TBF release mechanism. It can be seen that with delayed TBF release mechanism the number of (P)CCCH messages (Packet Uplink/Downlink Assignments) is significantly smaller. The difference is especially large in the e-mail test case. With normal TBF release mechanism the number of (P)CCCH messages is 671 whereas with delayed TBF release mechanism it is only 11. Also for netmeeting the the current release mechanism leads to a very high number of assignment messages. For RealPlayer the difference was not as large. Probably in the beginning RLC buffers become empty occasionally causing TBF to be released. However, after the situation has stabilized

and there is data in the RLC buffers all time then there is no difference whether normal or delayed release is used.

	<b>Mail</b>			
	Downlink		Uplink	
	Normal	Delayed	Normal	Delayed
# (P)CCCH messages	671	11	671	11
	<b>Realplayer streaming</b>			
	Downlink		Uplink	
	Normal	Delayed	Normal	Delayed
# (P)CCCH messages	19	3	22	3
	<b>Netmeeting</b>			
	Downlink		Uplink	
	Normal	Delayed	Normal	Delayed
# (P)CCCH messages	99	3	102	3

Table 1 Total number of (P)CCCH messages over the session

It was also tested how long time does it take to save the attachment in phase 7 and 9 of e-mail session. The results are shown in table 2.

	Delay	
Case	Normal	Timer
Attachment 1	6.5 min	1.5min
Attachment 2	10.5min	3.5min

Table 2. Saving time of an attachment (phase 7 and phase 9 of the example session)

With normal TBF release procedure the attachment saving time was 3-4 times longer than with delayed TBF release. This is a clear indication that the existing release mechanism is not suitable for interactive packet traffic.

## 5. CONCLUSIONS

It has been shown how some typical IP applications generate frequent interactive message exchanges between the client and the server. With existing TBF release mechanism this leads to a high number of TBF setups and releases which cause load on common control channels and increase the delay. From the test results it can be concluded that GPRS performance can be improved significantly by modifying the release mechanisms. This issue is seen as an important problem to solve as soon as possible.

## 6. REFERENCES

- [1] GSM 04.60 version 8.4.0 Release 1999.
- [2] W. Stevens. TCP/IP Illustrated, Volume 1. Addison-Wesley. 1994.

## 7. APPENDIX A. TRACE FROM E-MAIL SESSION<sup>4</sup>

```
Server:      * OK Microsoft Exchange IMAP4rev1 server version 5.5.2638.0
("server") ready

Client:      1 capability

Server:      * CAPABILITY IMAP4 IMAP4rev1 IDLE LITERAL+ LOGIN-REFERRALS MAILBOX-
REFERRALS NAMESPACE AUTH=NTLM
Server:      1 OK CAPABILITY completed.

Client:      2 login "loginname" "passwd"
Server:      2 OK LOGIN completed.

Client:      3 list "" "%"
Server:      * LIST () "/" Outbox

Client:      4 list "" "Outbox/%"
Server:      * LIST () "/" "Deleted Items"

Client:      5 list "" "Deleted Items/%"
Server:      * LIST (\Marked) "/" "Sent Items"

Client:      6 list "" "Sent Items/%"
Server:      * LIST () "/" Contacts

Client:      7 list "" "Contacts/%"
Server:      * LIST () "/" Notes

Client:      8 list "" "Notes/%"
Server:      * LIST () "/" Drafts

Client:      9 list "" "Drafts/%"
Server:      * LIST () "/" Journal

Client:      10 list "" "Journal/%"
Server:      * LIST (\Marked) "/" Calendar

Client:      11 list "" "Calendar/%"
Server:      * LIST () "/" Tasks

Client:      12 list "" "Tasks/%"

Server:      * LIST (\Marked) "/" INBOX

Client:      13 list "" "INBOX/%"
Server:      3 OK LIST completed.
Server:      4 OK LIST completed.
Server:      5 OK LIST completed.
Server:      6 OK LIST completed.
Server:      7 OK LIST completed.
Server:      8 OK LIST completed.
Server:      9 OK LIST completed.
Server:      10 OK LIST completed.
Server:      11 OK LIST completed.
Server:      12 OK LIST completed.
Server:      13 OK LIST completed.

Client:      14 list "" "INBOX"
Server:      * LIST (\Marked) "/" INBOX
Server:      14 OK LIST completed.
```

---

<sup>4</sup> Notice that login name, password, mail server address, user's mail address have been changed to "loginname", "passwd", "server", XXXX, YYYY.

```

Client:      15 select "INBOX"
Server:      * 5 EXISTS
Server:      * 1 RECENT
Server:      * FLAGS (\Seen \Answered \Flagged \Deleted \Draft)
Server:      * OK [PERMANENTFLAGS (\Seen \Answered \Flagged \Deleted \Draft)]

Server:      * OK [UIDVALIDITY 14556] UIDVALIDITY value.
Server:      15 OK [READ-WRITE] SELECT completed.

Client:      16 UID fetch 1:* (FLAGS)
Server:      * 1 FETCH (FLAGS (\Seen) UID 1749)
Server:      * 2 FETCH (FLAGS (\Seen) UID 1789)
Server:      * 3 FETCH (FLAGS (\Seen) UID 1807)
Server:      * 4 FETCH (FLAGS (\Seen) UID 1820)
Server:      * 5 FETCH (FLAGS (\Seen \Recent) UID 1852)
Server:      16 OK FETCH completed.

Client:      17 UID fetch 1852 (UID RFC822.SIZE RFC822.HEADER)
Server:      * 5 FETCH (UID 1852 RFC822.SIZE 444 RFC822.HEADER {417}
Server:      Received: by "server"
Server:      id <XXXXXX.YYYYYYY@"server">; Tue, 23 Nov 1999 16:57:39 +0200

Server:      Message-ID: <XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX@server>
Server:      From: "user's mail alias" <XXXXXX.YYYYYYY@nokia.com>
Server:      To: " user's mail alias " <XXXXXX.YYYYYYY@nokia.com>
Server:      Subject: IMAP
Server:      Date: Tue, 23 Nov 1999 16:57:39 +0200
Server:      MIME-Version: 1.0
Server:      Content-Type: text/plain;
Server:      charset="iso-8859-1"
Server:      )
Server:      17 OK FETCH completed.

Client:      18 UID fetch 1852 (UID RFC822.SIZE RFC822)
Server:      * 5 FETCH (UID 1852 RFC822.SIZE 444 RFC822 {444}
Server:      Received: by "server"
Server:      id < XXXXX.YYYYYYY@"server">; Tue, 23 Nov 1999 16:57:39 +0200

Server:      Message-ID: <XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX@server>
Server:      From: " user's mail alias " <XXXXXX.YYYYYYY @nokia.com>
Server:      To: " user's mail alias " <XXXXXX.YYYYYYY @nokia.com>
Server:      Subject: IMAP
Server:      Date: Tue, 23 Nov 1999 16:57:39 +0200
Server:      MIME-Version: 1.0
Server:      Content-Type: text/plain;
Server:      charset="iso-8859-1"
Server:      This is a test.
Server:      BR Petri
Server:      )
Server:      18 OK FETCH completed.

Client:      19 uid copy 1852 "Deleted Items"
Server:      19 OK COPY completed, 1 messages copied.

Client:      20 uid store 1852 +FLAGS (\Deleted)
Server:      * 5 FETCH (FLAGS (\Seen \Deleted \Recent) UID 1852)
Server:      20 OK STORE completed.

Client:      21 fetch 5 (UID)
Server:      * 5 FETCH (UID 1852)
Server:      21 OK FETCH completed.

```



```

Client:      22 fetch 5 (UID)
Server:      * 6 EXISTS
Server:      * 2 RECENT
Server:      * 5 FETCH (UID 1852)
Server:      22 OK FETCH completed.

Client:      23 noop
Server:      23 OK NOOP completed.

Client:      24 UID fetch 1:* (FLAGS)
Server:      * 1 FETCH (FLAGS (\Seen) UID 1749)
Server:      * 2 FETCH (FLAGS (\Seen) UID 1789)
Server:      * 3 FETCH (FLAGS (\Seen) UID 1807)
Server:      * 4 FETCH (FLAGS (\Seen) UID 1820)
Server:      * 5 FETCH (FLAGS (\Seen \Deleted \Recent) UID 1852)
Server:      * 6 FETCH (FLAGS (\Recent) UID 1853)
Server:      24 OK FETCH completed.

Client:      25 UID fetch 1853 (UID RFC822.SIZE RFC822.HEADER)
Server:      * 6 FETCH (UID 1853 RFC822.SIZE 459 RFC822.HEADER {424}
Server:      Received: by "server"
Server:      id <XXXXX.YYYYYYY@server>; Tue, 23 Nov 1999 18:31:49 +0200

Server:      Message-ID: <XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX@server>
Server:      From: " user's mail alias " <XXXXX.YYYYYY@nokia.com>
Server:      To: " user's mail alias " <XXXXX.YYYYYY@nokia.com>
Server:      Subject: New message
Server:      Date: Tue, 23 Nov 1999 18:31:49 +0200
Server:      MIME-Version: 1.0
Server:      Content-Type: text/plain;
Server:      charset="iso-8859-1"
Server:      )
Server:      25 OK FETCH completed.

Client:      xxxx logout

```