LLD +CNMA bugfix

Project	G23-GPRS Protocol Stack
Document Type	Detailed Specification
Title	LLD +CNMA bugfix
Author	TI Employee
Creation Date	29 January, 2004
Last Modified	29 January, 2004
ID and Version	8462.726.04.001
Status	Being Processed

0 Document Control

0.1 Document History

ID	Author	Date	Status
8462.726.04.001	Thomas Schott	29 January, 2004	Being Processed

0.2 References, Abbreviations, Terms

[TI 7010.801] 7010.801, References and Vocabulary, Texas Instruments.

Table of Contents

1 Introduction

The present implementation (13.01.2004) of +CNMA is incomplete regarding the GSM07.05 standard.
Some features that described in GSM07.05 are missing. All described features, which are described in chapter 2.1 in bold letter must be provide too.

2 Overview

2.1 'New Message Acknowledgement to ME/TA' +CNMA in Text Mode

Action Command Syntax

Command	Possible response(s)
if text mode (+CMGF=1): +CNMA	*+CMS ERROR: <err>*
+CNMA=?	

Description
Execution command confirms correct reception of a new message (SMS-DELIVER or SMS-STATUS-REPORT) which is routed directly to the TE (refer command +CNMI tables **Error! Bookmark not defined.** and **Error! Bookmark not defined.**).

This acknowledgement command (causing ME to send RP-ACK to the network) shall be used when +CSMS parameter <service> equals 1. TA shall not send another +CMT or +CDS result code to TE before previous one is acknowledged.

If ME does not get acknowledgement within required time (network timeout), ME should send RP-ERROR to the network. **ME/TA shall automatically disable routing to TE by setting both <mt> and <ds> values of +CNMI to zero.**

If command is executed, but no acknowledgement is expected, or some other ME related error occurs, final result code +CMS ERROR: <err> is returned. See chapter Message Service Failure Result Code for a list of <err> values.

NOTE: In case that a directly routed message must be buffered in ME/TA (possible when +CNMI parameter <mode> equals 0 or 2) or AT interpreter remains too long in a state where result codes cannot be sent to TE (e.g. user is entering a message using +CMGS), **acknowledgement (RP-ACK) must be sent to the network without waiting +CNMA command from TE. Later, when buffered result codes are flushed to TE, TE must send +CNMA acknowledgement for each result code. In this way, ME/TA can determine if message should be placed in non-volatile memory and routing to TE disabled (+CNMA not received).** Refer command +CNMI for more details how to use <mode> parameter reliably.

Conclusion:
- aus 1* folgt, dass das Rücksetzen von <mt> und <ds> noch nachimplementiert werden muss.
- aus 2* folgt, dass die SMS-entity bei eingestelltem <mode> == ‚0' oder ‚2' selbst das RP-ACK zum Netz schicken muss. Da die SMS-entity den eingestellten mode nicht weiss, muss in diesem Fall der ACI diese Bestätigung zur SMS-Entity generieren. Der ACI erwartet jedoch weiterhin von der Applikation die +CNMA Nachricht, wenn diese dann aus dem Puffer an die Applikation gesendet wird. Erst dann wird die nächste gespeicherte SMS an die Applikation gesendet.
  Wichtig: Sollte bei diesem Szenario das Acknowledgement für eine gespeicherte SMS ausfallen, so muß auch in diesem Fall <mt> und <ds> auf ‚0' gesetzt werden (incl. Configure_request zur SMS-entity) und zusätzlich alle noch im ACI Puffer verbleibenden SMS's per +CMGW auf die SIM oder in das ME geschrieben werden.

2.2 'New Message Acknowledgement to ME/TA' +CNMA in PDU Mode

Action Command Syntax

Command	Possible response(s)
if PDU mode (+CMGF=0): +CNMA[=<n>[,<length>[<CR>***PDU is given***<ctrl-Z/ESC>]]]	*+CMS ERROR: <err>*
+CNMA=?	**if PDU mode (+CMGF=0):** +CNMA: (list of supported <n>s)

Description
Execution command confirms reception of a new message (SMS-DELIVER or SMS-STATUS-REPORT) which is routed directly to the TE (refer command +CNMI tables **Error! Bookmark not defined.** and **Error! Bookmark not defined.**). This acknowledgement command shall be used when +CSMS parameter <service> equals 1. In PDU mode, it is possible to send either positive (RP-ACK) or negative (RP-ERROR) acknowledgement to the network. Parameter <n> defines which one will be sent. Optionally (when <length> is greater than zero) an acknowledgement TPDU (SMS-DELIVER-REPORT for RP-ACK or RP-ERROR) may be sent to the network. The entering of PDU is done similarly as specified in command Send Message +CMGS, except that the format of <ackpdu> is used instead of <pdu> (i.e. SMSC address field is not present). PDU shall not be bounded by double quotes. TA shall not send another +CMT or +CDS result code to TE before previous one is acknowledged.

If ME does not get acknowledgement within required time (network timeout), ME should send RP-ERROR to the network. ME/TA shall automatically disable routing to TE by setting both <mt> and <ds> values of +CNMI to zero.
If command is executed, but no acknowledgement is expected, or some other ME related error occurs, final result code +CMS ERROR: <err> is returned. See chapter Message Service Failure Result Code for a list of <err> values.

NOTE: In case that a directly routed message must be buffered in ME/TA (possible when +CNMI parameter <mode> equals 0 or 2) or AT interpreter remains too long in a state where result codes cannot be sent to TE (e.g. user is entering a message using +CMGS), acknowledgement (RP-ACK) must be sent to the network without waiting +CNMA command from TE. Later, when buffered result codes are flushed to TE, TE must send +CNMA[=0] acknowledgement for each result code. In this way, ME/TA can determine if message should be placed in non-volatile memory and routing to TE disabled (+CNMA[=0] not received). Refer command +CNMI for more details how to use <mode> parameter reliably.

Test command returns a list of supported <n> values. If the only value supported is 0, the device does not support sending of TPDU.

Defined Values
<n>:
0 command operates similarly as defined for the text mode
1 send RP-ACK (or buffered result code received correctly)
2 send RP-ERROR (if PDU is not given, ME/TA shall send SMS-DELIVER-REPORT with GSM 03.40 TP-FCS value set to 'FF' (unspecified error cause))

3 Message sequence charts

First of all, the network wants a +CNMA acknowledgement if the AT commands are compatible with GSM07.05 Phase 2+. This can be selected by setting the +CSMS service to '1'.
Furthermore the behavior of the SMS receiving process depends on +CNMI settings. For example you can choose if the received SMS will be buffered or not.

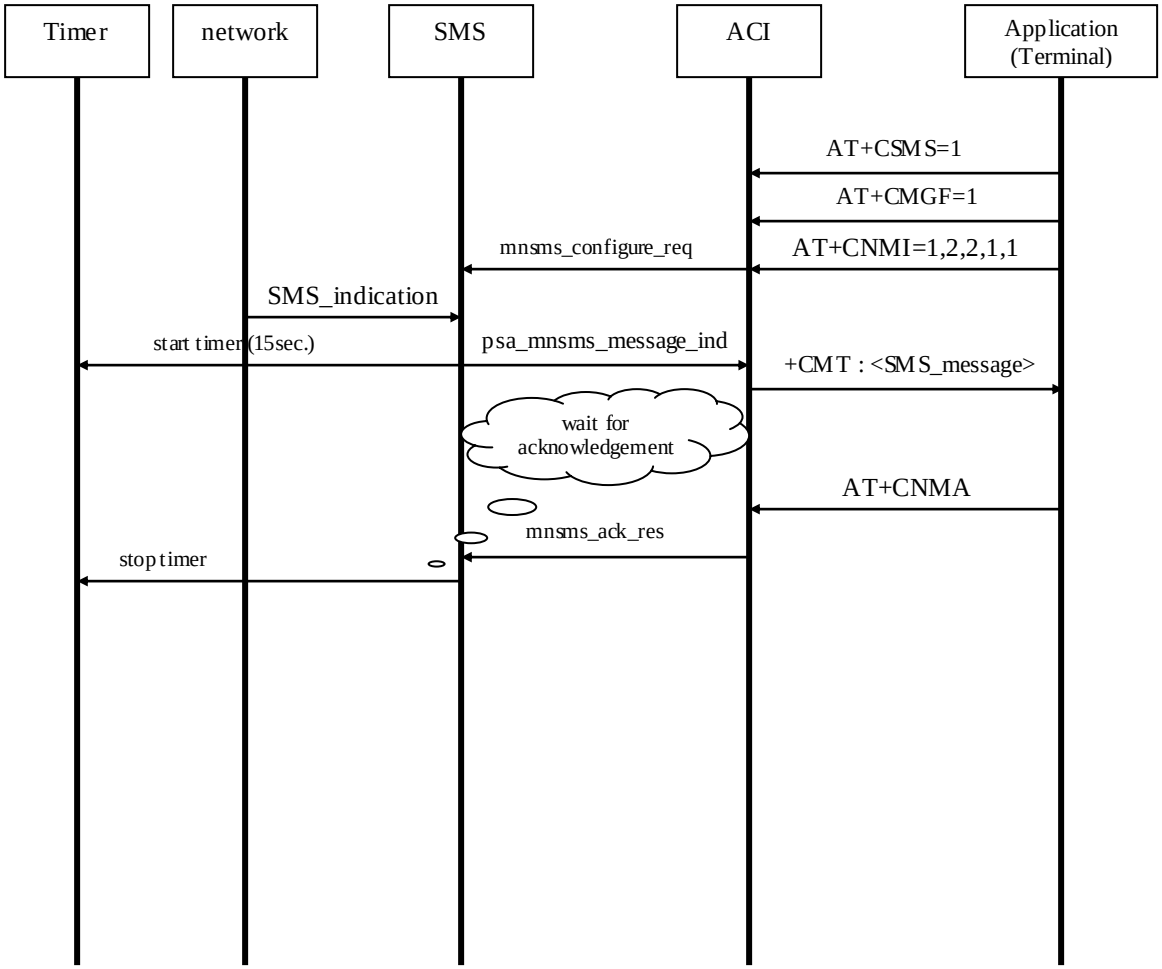
3.1 Text Mode

3.1.1 SMS receive without buffering

3.1.1.1 Acknowledgement of one received SMS

This scenario describes the following behavior:

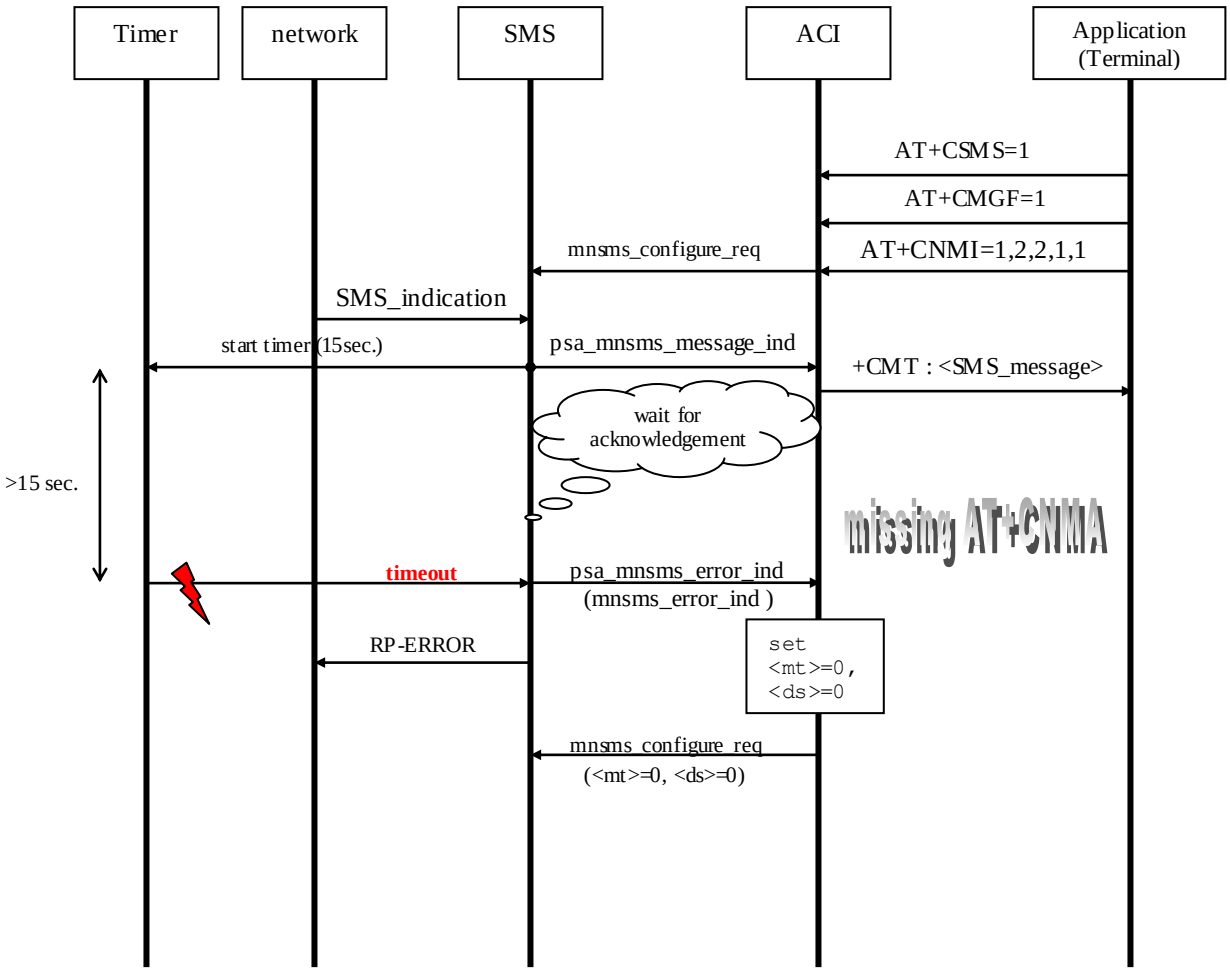
- simple SMS-acknowledgement
- no buffering in ACI (+CNMI <mode> not equal ‚0' or ‚2')
- no timeout
- no failure
- only one SMS in text mode (AT+CMGF=1) received
- Phase 2+ SMS service (AT+CSMS=1)



3.1.1.2 Acknowledgement of one received SMS with timeout

This scenario describes the following behavior:

- simple SMS-acknowledgement
- no buffering in ACI (+CNMI <mode> not equal ‚0' or ‚2')
- **timeout (SMS entity waits 15 sec. for AT+CNMA)**
- no failure
- only one SMS in text mode (AT+CMGF=1) received
- Phase 2+ SMS service (AT+CSMS=1)

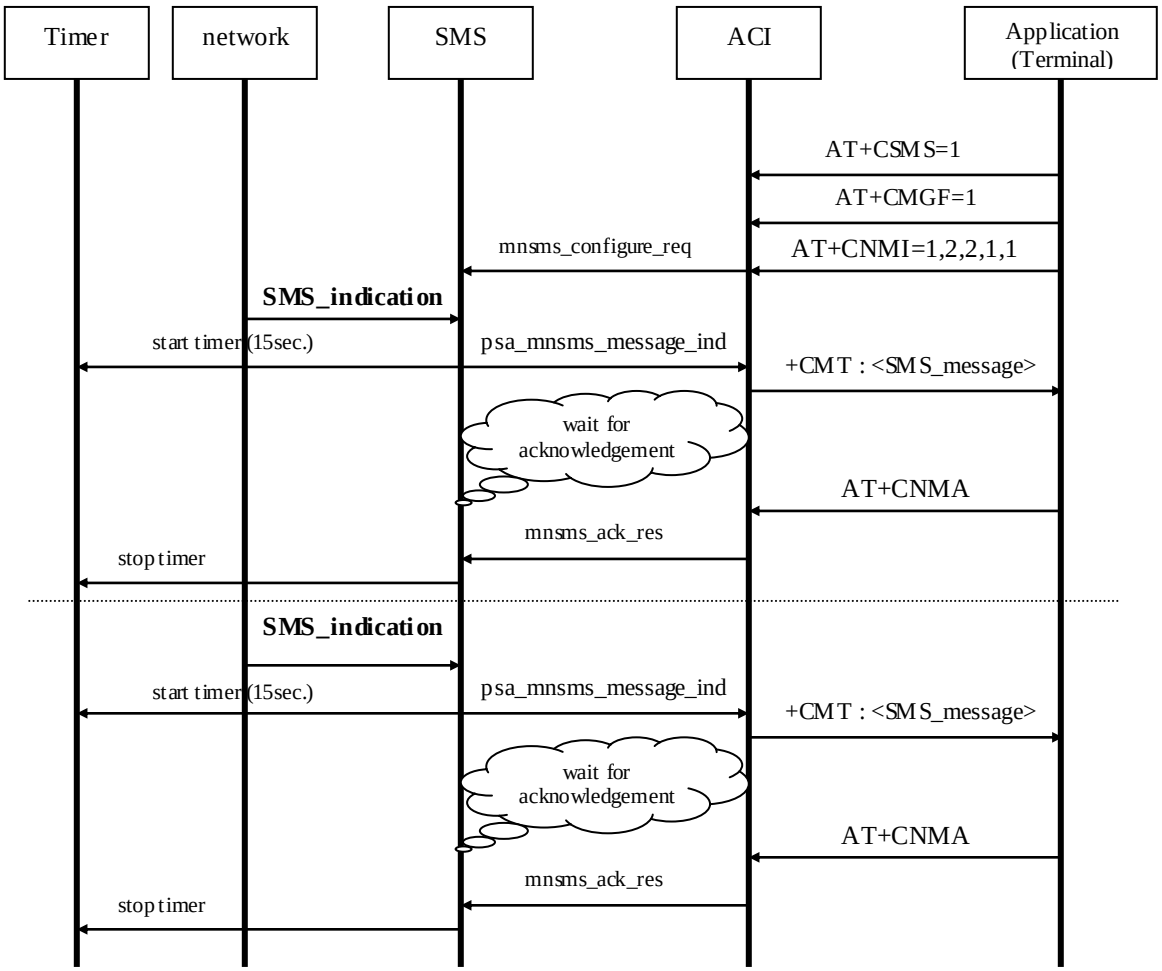


3.1.1.3 Acknowledgement of multiple received SMS

If more than one SMS received the ACI will expect a AT+CNMA for each message. In this case the timer will be restart after each message.

This scenario describes the following behavior:

- simple SMS-acknowledgement
- no buffering in ACI (+CNMI <mode> not equal ‚0' or ‚2')
- no timeout
- no failure
- more than one SMS in text mode (AT+CMGF=1) received
- Phase 2+ SMS service (AT+CSMS=1)



3.1.2 SMS receive with buffering

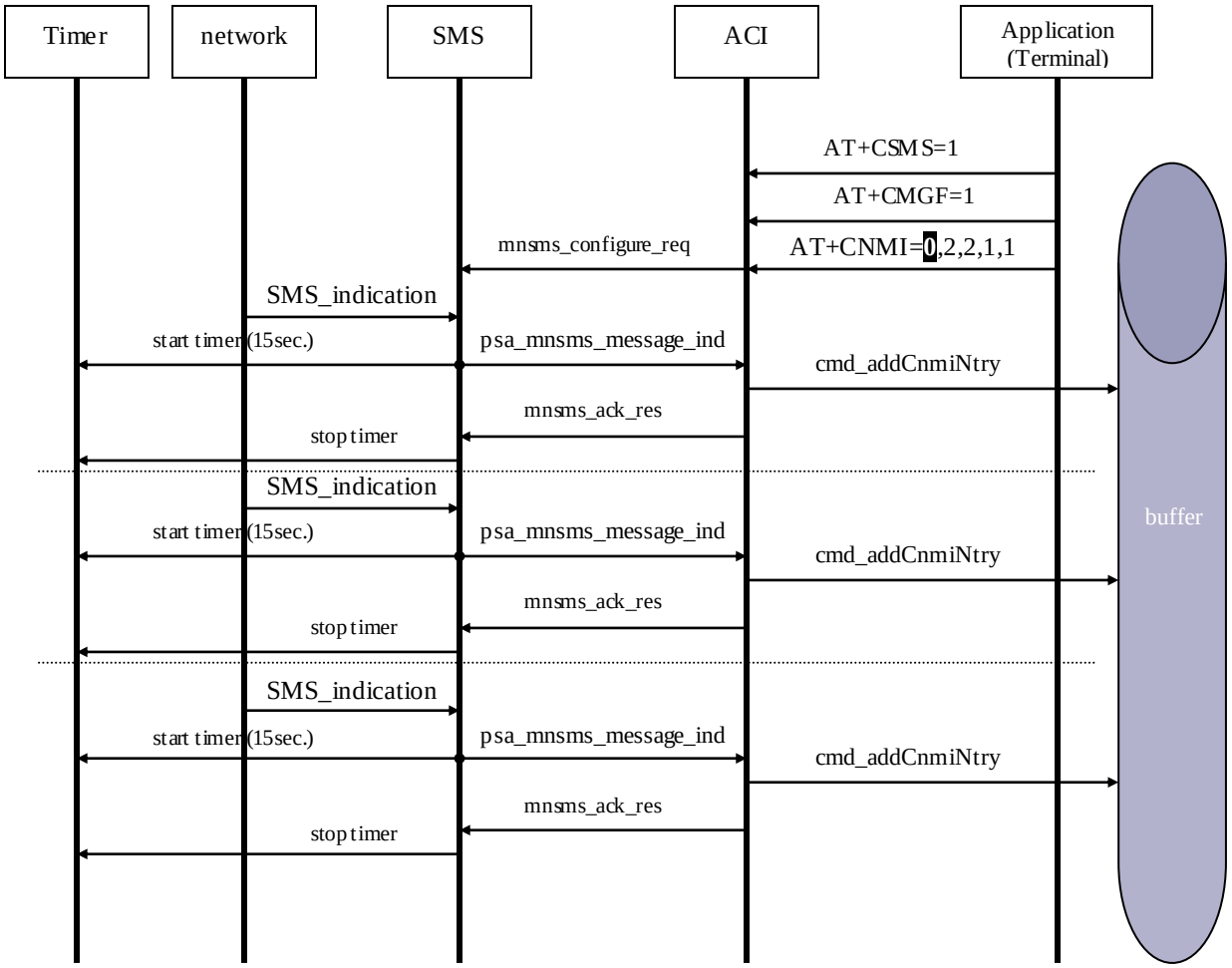
If the <mode> parameter of +CNMI equals '0' or '2' (data mode) the received SMS has to be buffered in a volatile memory space. This buffer has free space for four SMS entries.

3.1.2.1 Acknowledgement of buffered received SMS

This scenario describes the following behavior:

- simple SMS-acknowledgement
- buffering in ACI (+CNMI <mode> not equal ‚0' or ‚2')
- no timeout
- no failure
- SMS text mode (AT+CMGF=1)
- Phase 2+ SMS service (AT+CSMS=1)

In this scenario, the ACI will be store each message in the buffer. If the buffer is full, the oldest indications will be discarded and replaced with the new indications. After this the ACI sends the acknowledge response to the SMS entity to simulate that the "user" has seen this incoming messages.

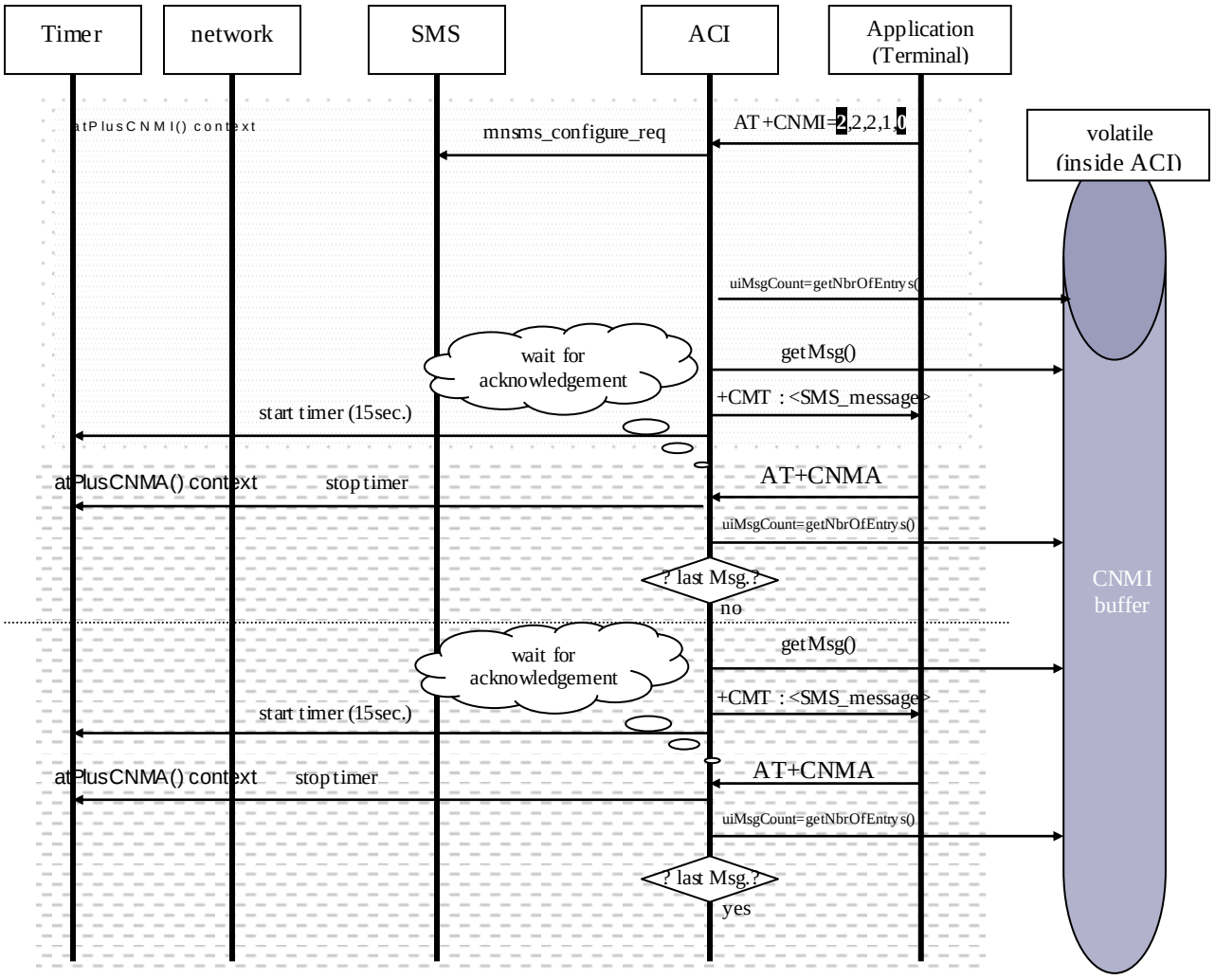


3.1.2.2 Flushing of buffered SMS (switch from data- to command mode)

This scenario describes the following behavior:

- simple SMS-acknowledgement
- buffering in ACI (+CNMI <mode> not equal ‚0' or ‚2')
- no timeout
- no failure
- SMS text mode (AT+CMGF=1)
- Phase 2+ SMS service (AT+CSMS=1)
- flushing the buffered SMS by AT+CNMI=**2**,2,2,1,**0**

Received SMS has to be buffered if the application is not be able to receive this. The buffered SMS can be send when the application is ready to receive or when the <bfr> setting of +CNMI AT command is set to '0'. When the buffer flushed, all relayed entries will be deleting. Each received message must be acknowledged via AT+CNMA command if the Phase 2+ SMS service mode is set. If the Phase 2+ SMS service mode is not set, the behavior is the same except the AT+CNMA command must not send and all SMS will be delivered o the application at once.



3.1.2.3 Flushing of buffered SMS with time out

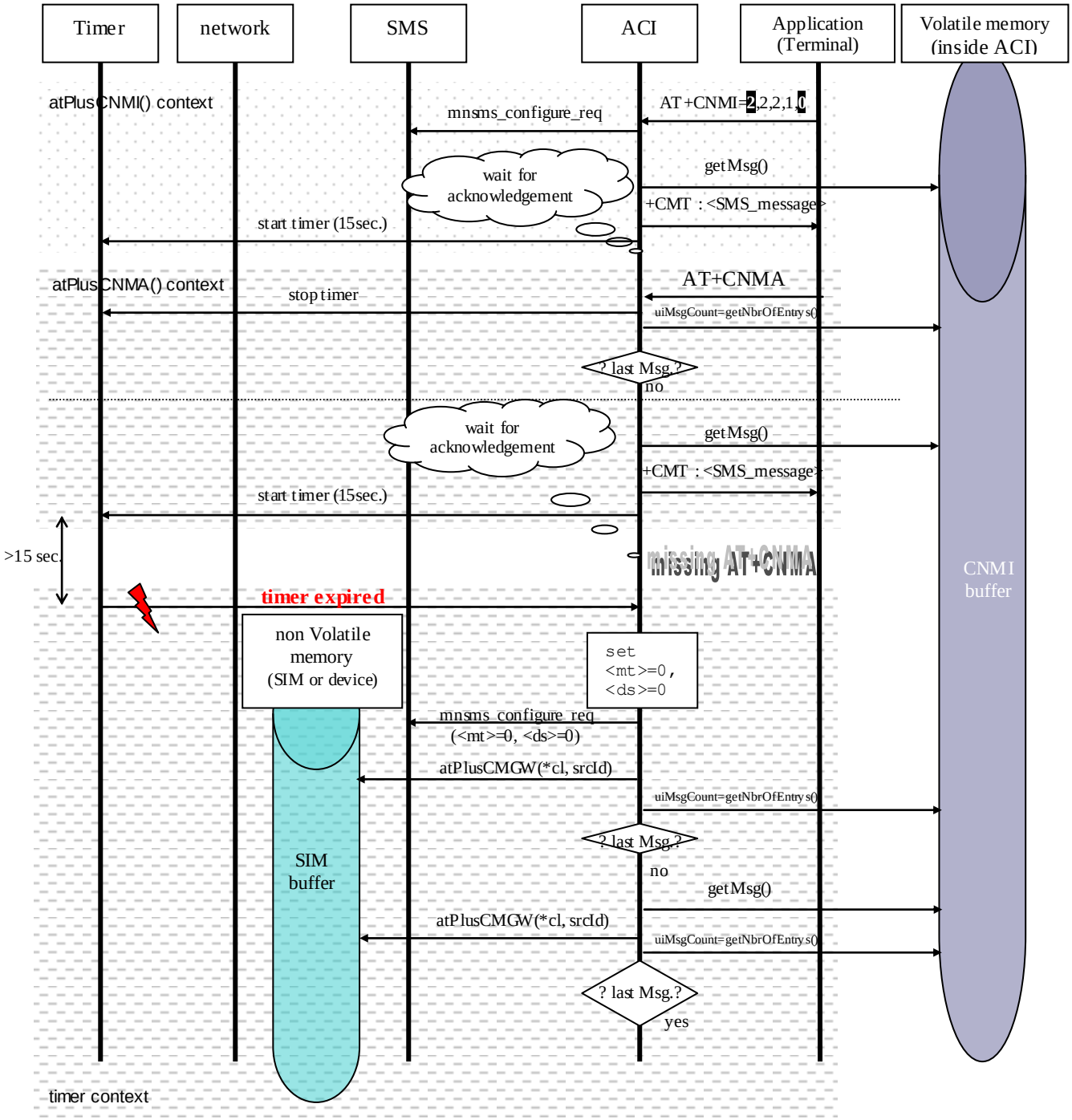
This scenario describes the following behavior:

- simple SMS-acknowledgement
- buffering in ACI (+CNMI <mode> not equal ‚0' or ‚2')
- no timeout
- no failure
- SMS text mode (AT+CMGF=1)
- Phase 2+ SMS service (AT+CSMS=1)
- flushing the buffered SMS by AT+CNMI=**2**,2,2,1,**0**

Received SMS has to be buffered if the application is not be able to receive it. The buffered SMS can be send when the application is ready to receive or when the <bfr> setting of +CNMI AT command is set to '0'. When the buffer flushed, all relayed entries will be deleting. Normally each received message must be acknowledged via AT+CNMA command if the Phase 2+ SMS service mode is set. If the Phase 2+ SMS service mode is not set, the behavior is the same except the AT+CNMA command must not send and all SMS will be delivered o the application at once.

Now, in our picture below after the second relayed message, the application does not send the +CNMA because it is in state DATA-MODE or busy for other reasons. In this case a timer (15sec.) will be expired and all other remained messages will be stored into the non-volatile memory (in SIM or MS) with status "unread" and the +CNMI parameter <ds> and <mt> will be reset to '0'.

In the following scenario a content of three SMS in the buffer will be considered. The first SMS will be delivered successful. The second SMS cannot be received from the application (AT command interpreter). Thus, the third SMS has to be stored into the non-volatile memory.


Timer
network
SMS
ACI
Application (Terminal)
Volatile memory (inside ACI)
atPlusCNMI() context
mnsms_configure_req
AT+CNMI=2,2,2,1,0
wait for acknowledgement
getMsg()
+CMT : <SMS_message>
start timer (15sec.)
atPlusCNMA() context
stop timer
AT+CNMA
uiMsgCount=getNbrOfEntrys()
? last Msg.?
no
wait for acknowledgement
getMsg()
+CMT : <SMS_message>
start timer (15sec.)
>15 sec.
missing AT+CNMA
CNMI buffer
timer expired
non Volatile memory (SIM or device)
set <mt>=0, <ds>=0
mnsms_configure_req (<mt>=0, <ds>=0)
atPlusCMGW(*cl, srcId)
uiMsgCount=getNbrOfEntrys()
? last Msg.?
SIM buffer
no
getMsg()
atPlusCMGW(*cl, srcId)
uiMsgCount=getNbrOfEntrys()
? last Msg.?
yes
timer context

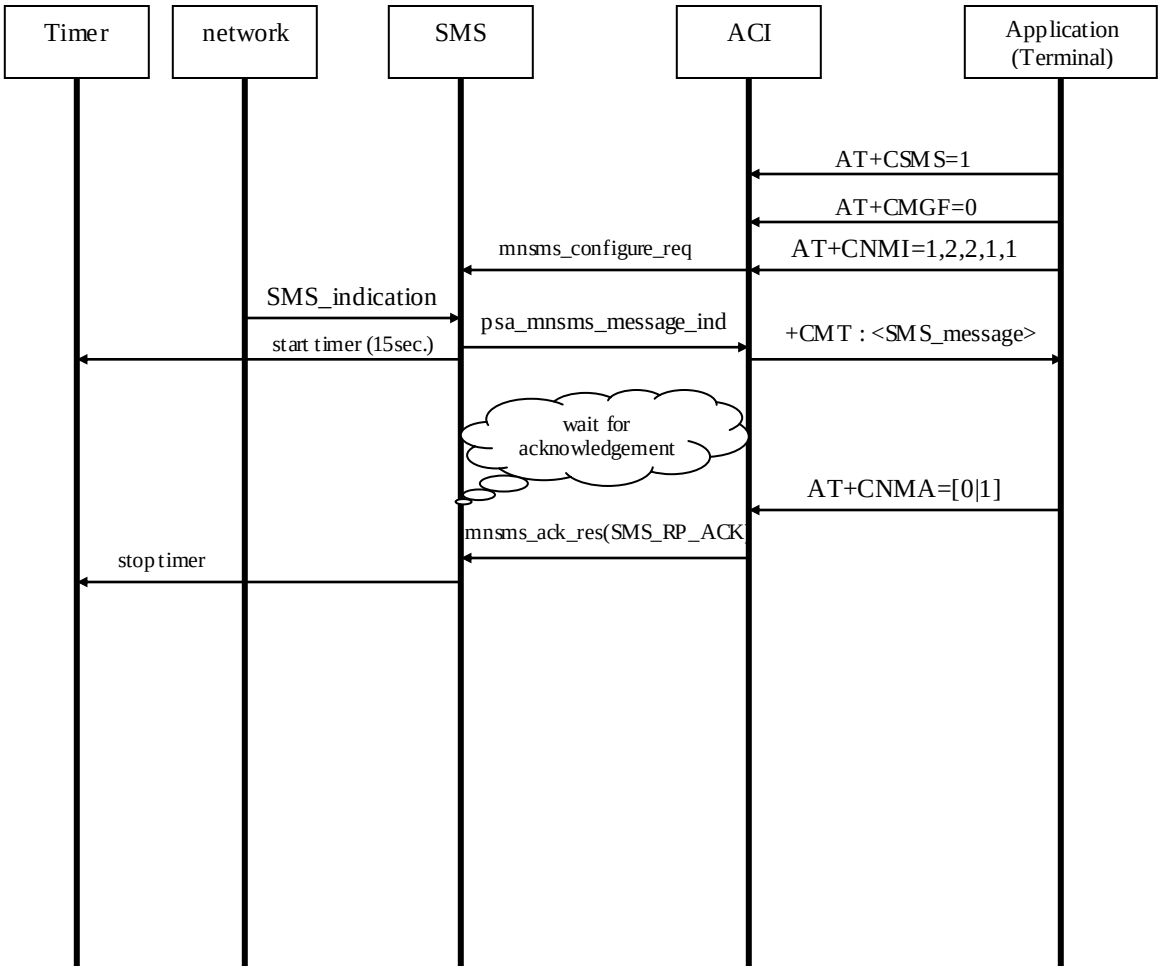
3.2 PDU Mode

3.2.1 SMS receive without buffering

3.2.1.1 Positive acknowledgement of one received SMS

This scenario describes the following behavior:

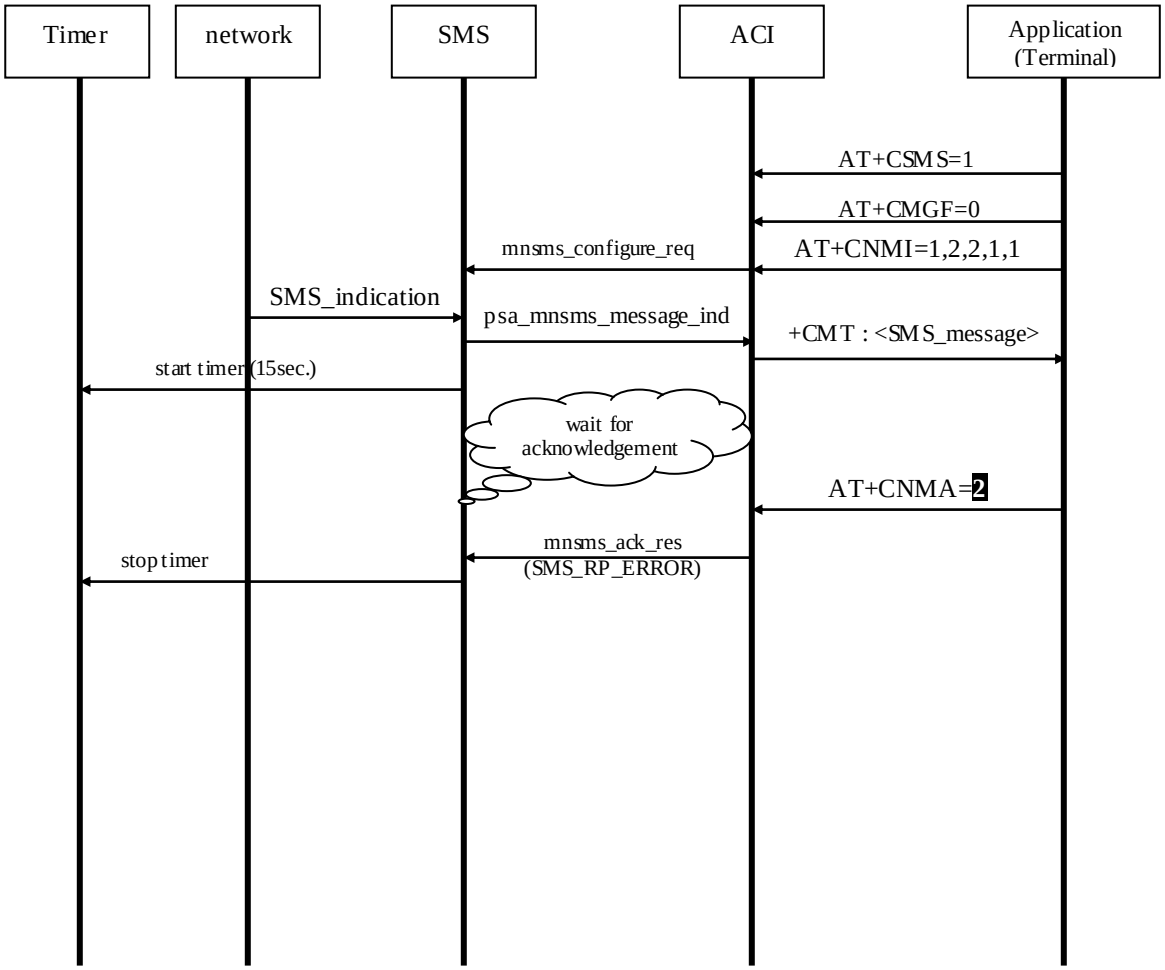
- simple SMS-acknowledgement
- no buffering in ACI (+CNMI <mode> not equal ‚0' or ‚2')
- no timeout
- no failure
- only one SMS in PDU mode (AT+CMGF=0) received
- Phase 2+ SMS service (AT+CSMS=1)



3.2.1.2 Negative acknowledgement of one received SMS

This scenario describes the following behavior:

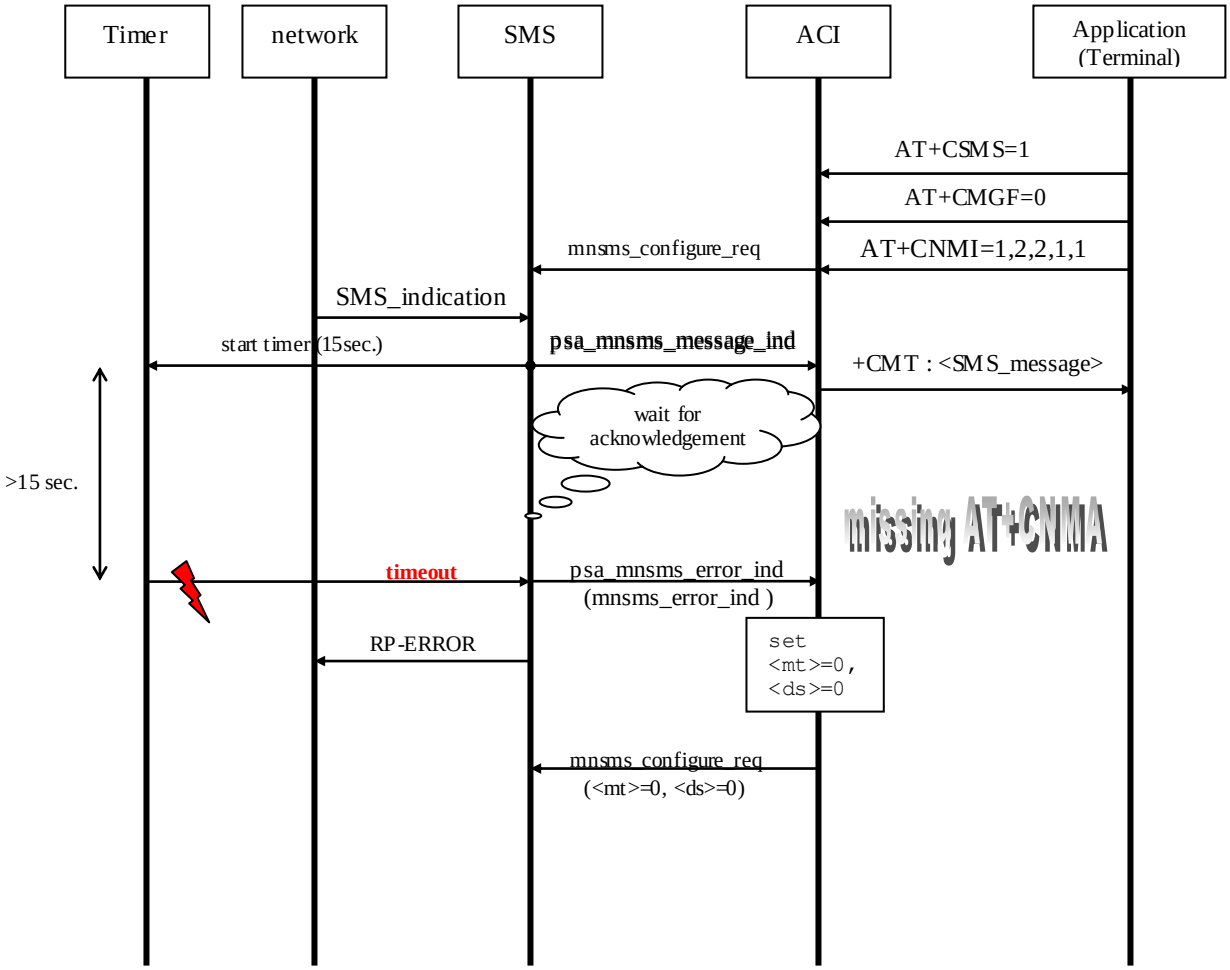
- simple negative SMS-acknowledgement
- no buffering in ACI (+CNMI <mode> not equal ‚0' or ‚2')
- no timeout
- no failure
- only one SMS in PDU mode (AT+CMGF=0) received
- Phase 2+ SMS service (AT+CSMS=1)



3.2.1.3 Acknowledgement of one received SMS with timeout

This scenario describes the following behavior:

- simple SMS-acknowledgement
- no buffering in ACI (+CNMI <mode> not equal ‚0' or ‚2')
- **timeout (SMS entity waits 15 sec. for AT+CNMA)**
- only one SMS in PDU mode (AT+CMGF=0) received
- Phase 2+ SMS service (AT+CSMS=1)

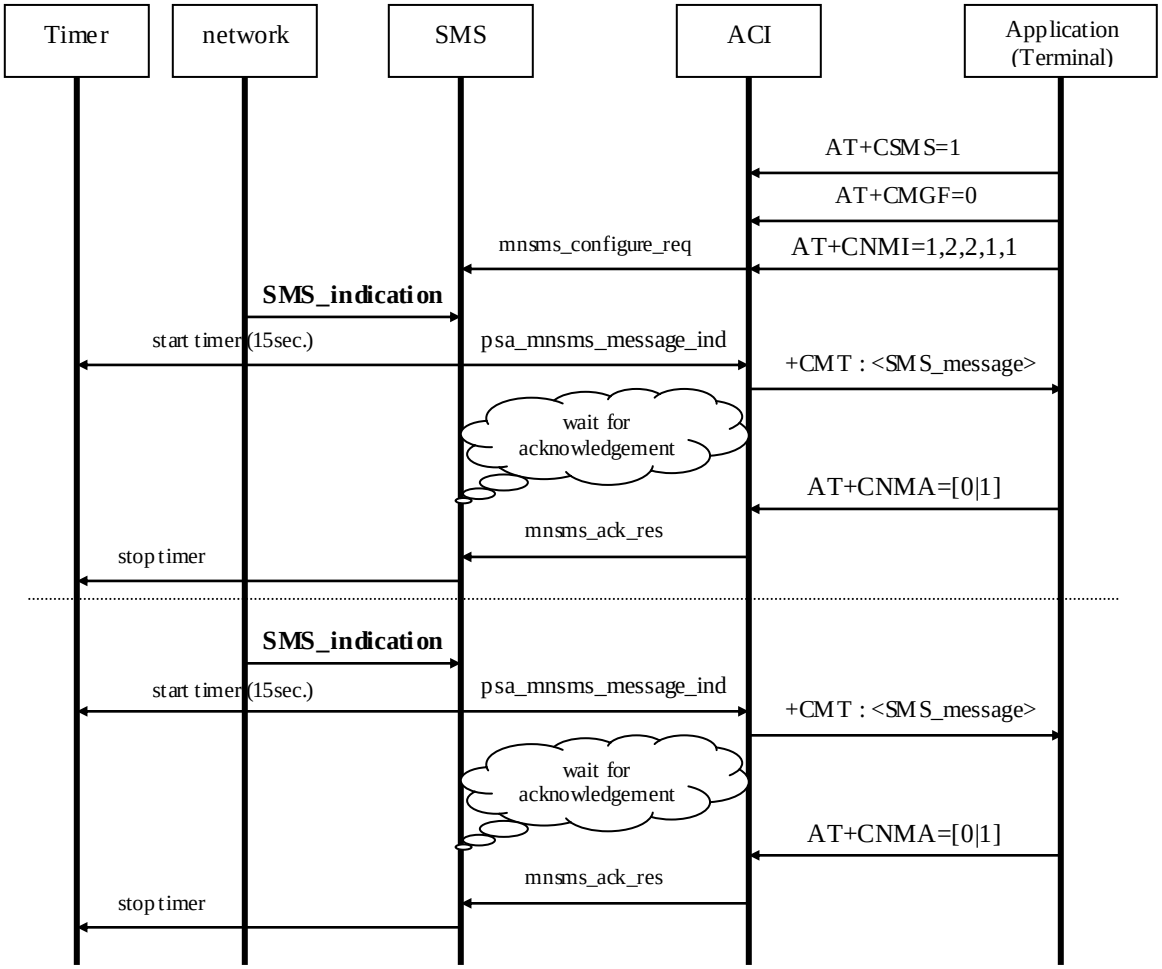


3.2.1.4 Acknowledgement of multiple received SMS

If more than one SMS received the ACI will expect a AT+CNMA for each message. In this case the timer will be restart after each message.

This scenario describes the following behavior:

- simple SMS-acknowledgement
- no buffering in ACI (+CNMI <mode> not equal ‚0' or ‚2')
- no timeout
- no failure
- more than one SMS in text mode (AT+CMGF=0) received
- Phase 2+ SMS service (AT+CSMS=1)



3.2.2 SMS receive with buffering

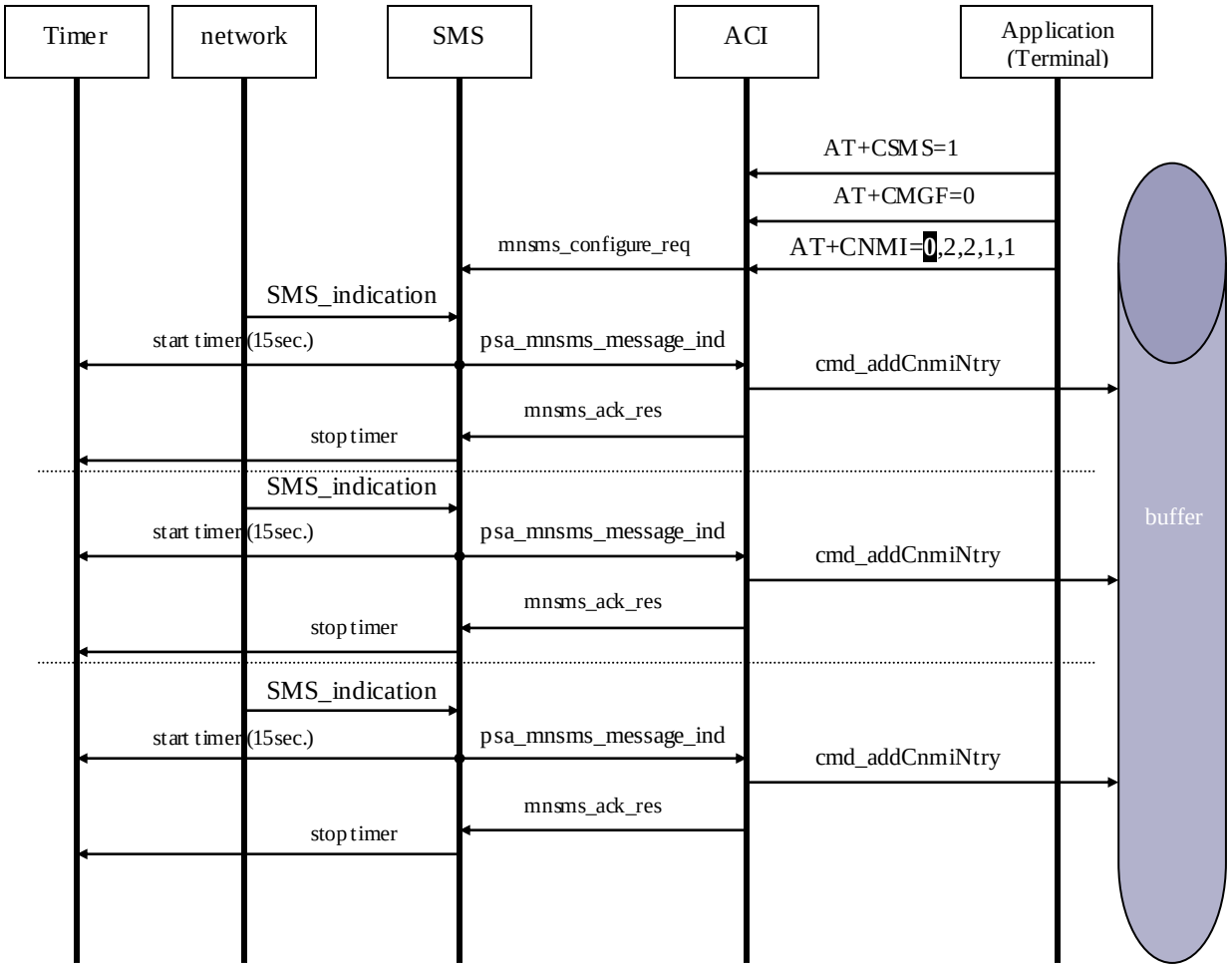
If the <mode> parameter of +CNMI equals '0' or '2' (data mode) the received SMS has to be buffered in a volatile memory space. This buffer has free space for four SMS entries.

3.2.2.1 Acknowledgement of buffered received SMS

This scenario describes the following behavior:

- simple SMS-acknowledgement
- buffering in ACI (+CNMI <mode> not equal ‚0' or ‚2')
- no timeout
- no failure
- SMS text mode (AT+CMGF=0)
- Phase 2+ SMS service (AT+CSMS=1)

In this scenario, the ACI will be stored each message in a buffer. If the buffer is full, the oldest indications will be discarded and replaced with the new indications. After this the ACI sends the acknowledge response to the SMS entity to simulate that the "user" has seen this incoming messages.

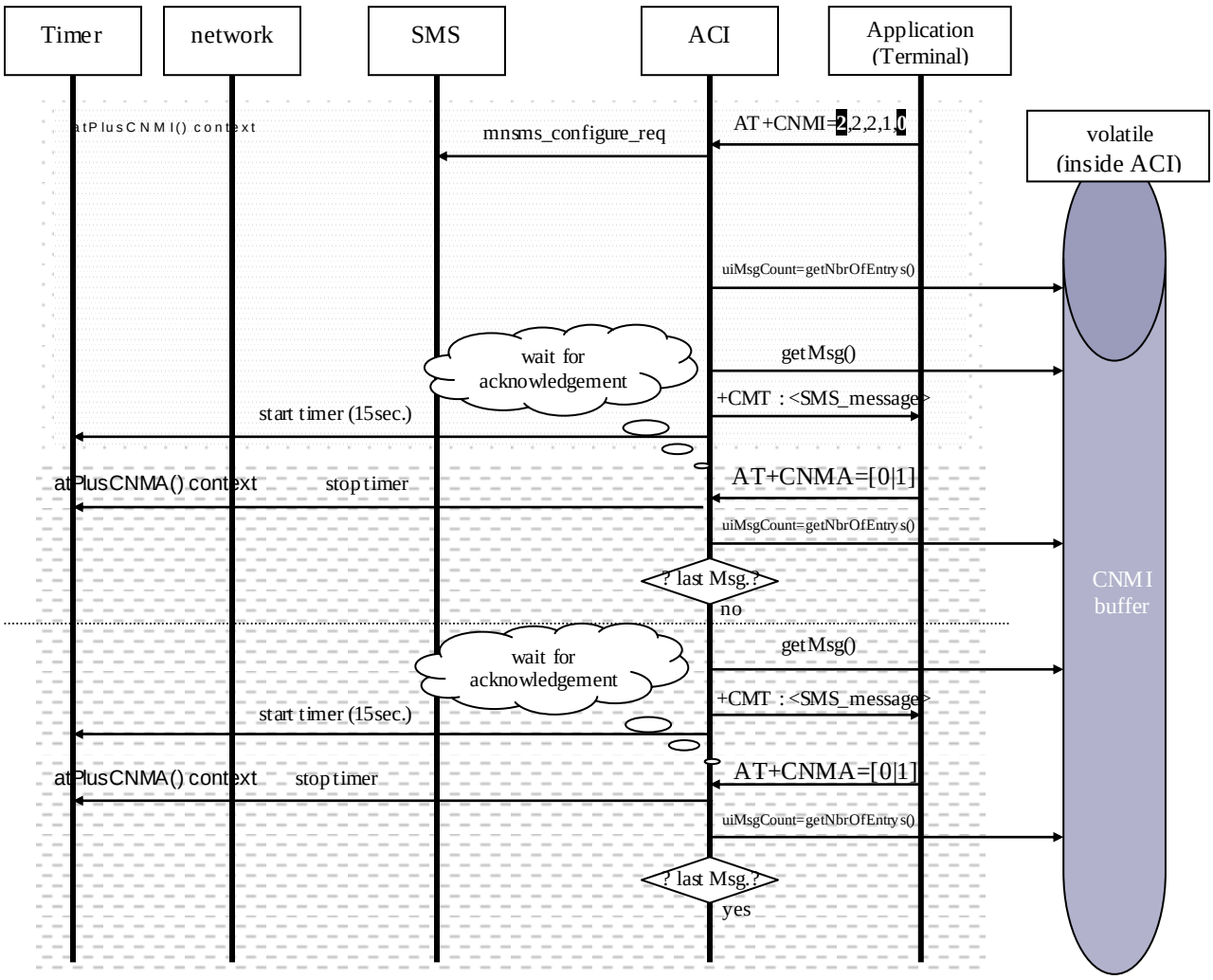


3.2.2.2 Flushing of buffered SMS

This scenario describes the following behavior:

- simple SMS-acknowledgement
- buffering in ACI (+CNMI <mode> not equal ‚0' or ‚2')
- no timeout
- no failure
- SMS text mode (AT+CMGF=0)
- Phase 2+ SMS service (AT+CSMS=1)
- flushing the buffered SMS by AT+CNMI=**2**,2,2,1,**0**

Received SMS has to be buffered if the application is not be able to receive this. The buffered SMS can be send when the application is ready to receive or when the <bfr> setting of +CNMI AT command is set to '0'. When the buffer flushed, all relayed entries will be deleting. Each received message must be acknowledged via AT+CNMA command if the Phase 2+ SMS service mode is set. If the Phase 2+ SMS service mode is not set, the behavior is the same except the AT+CNMA command must not send and all SMS will be delivered o the application at once.



3.2.2.3 Flushing of buffered SMS with time out

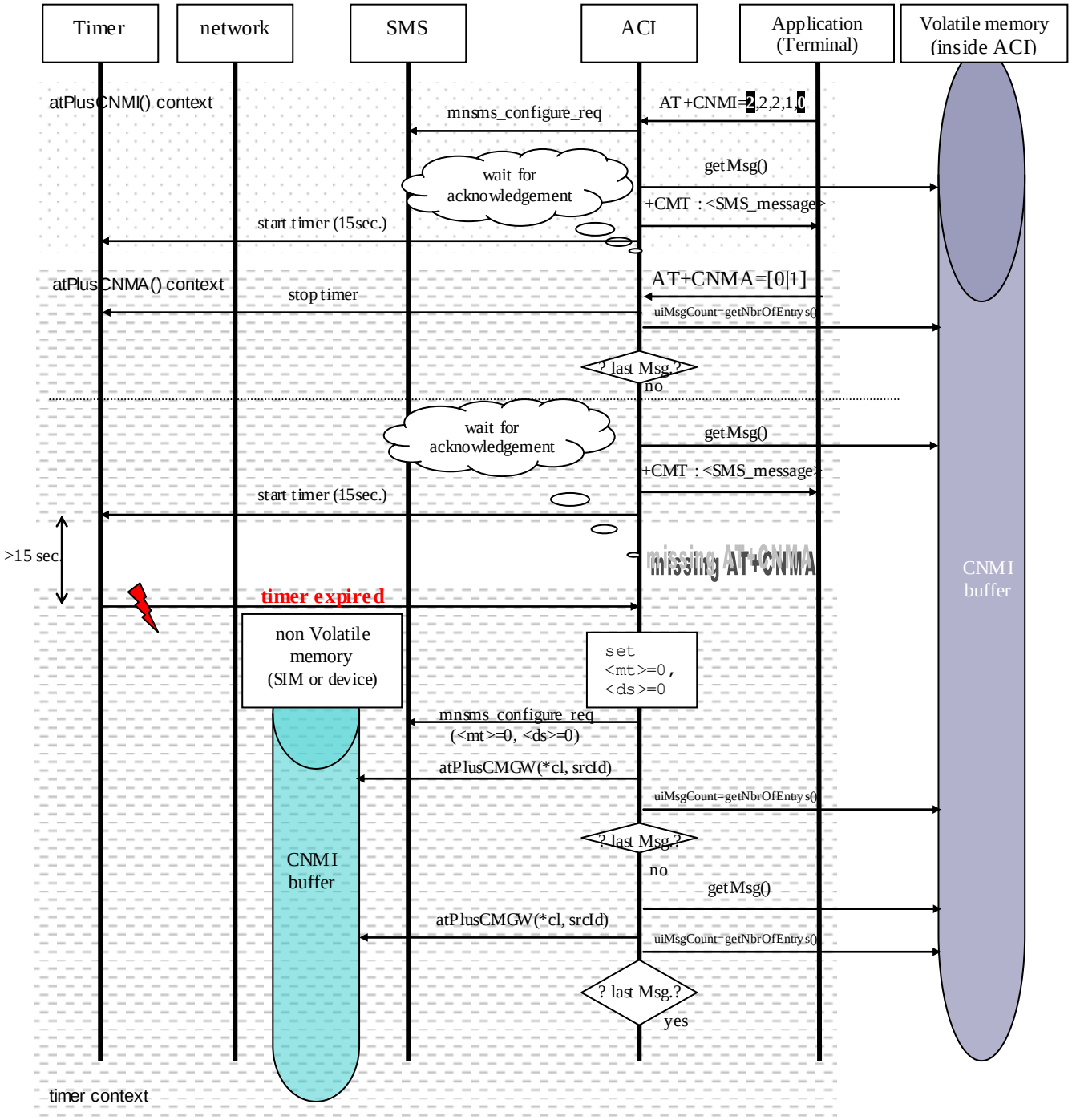
This scenario describes the following behavior:

- simple SMS-acknowledgement
- buffering in ACI (+CNMI <mode> not equal ‚0' or ‚2')
- no timeout
- no failure
- SMS text mode (AT+CMGF=0)
- Phase 2+ SMS service (AT+CSMS=1)
- flushing the buffered SMS by AT+CNMI=**2**,2,2,1,**0**

Received SMS has to be buffered if the application is not be able to receive it. The buffered SMS can be send when the application is ready to receive or when the <bfr> setting of +CNMI AT command is set to '0'. When the buffer flushed, all relayed entries will be deleting. Normally each received message must be acknowledged via AT+CNMA command if the Phase 2+ SMS service mode is set. If the Phase 2+ SMS service mode is not set, the behavior is the same except the AT+CNMA command must not send and all SMS will be delivered o the application at once.

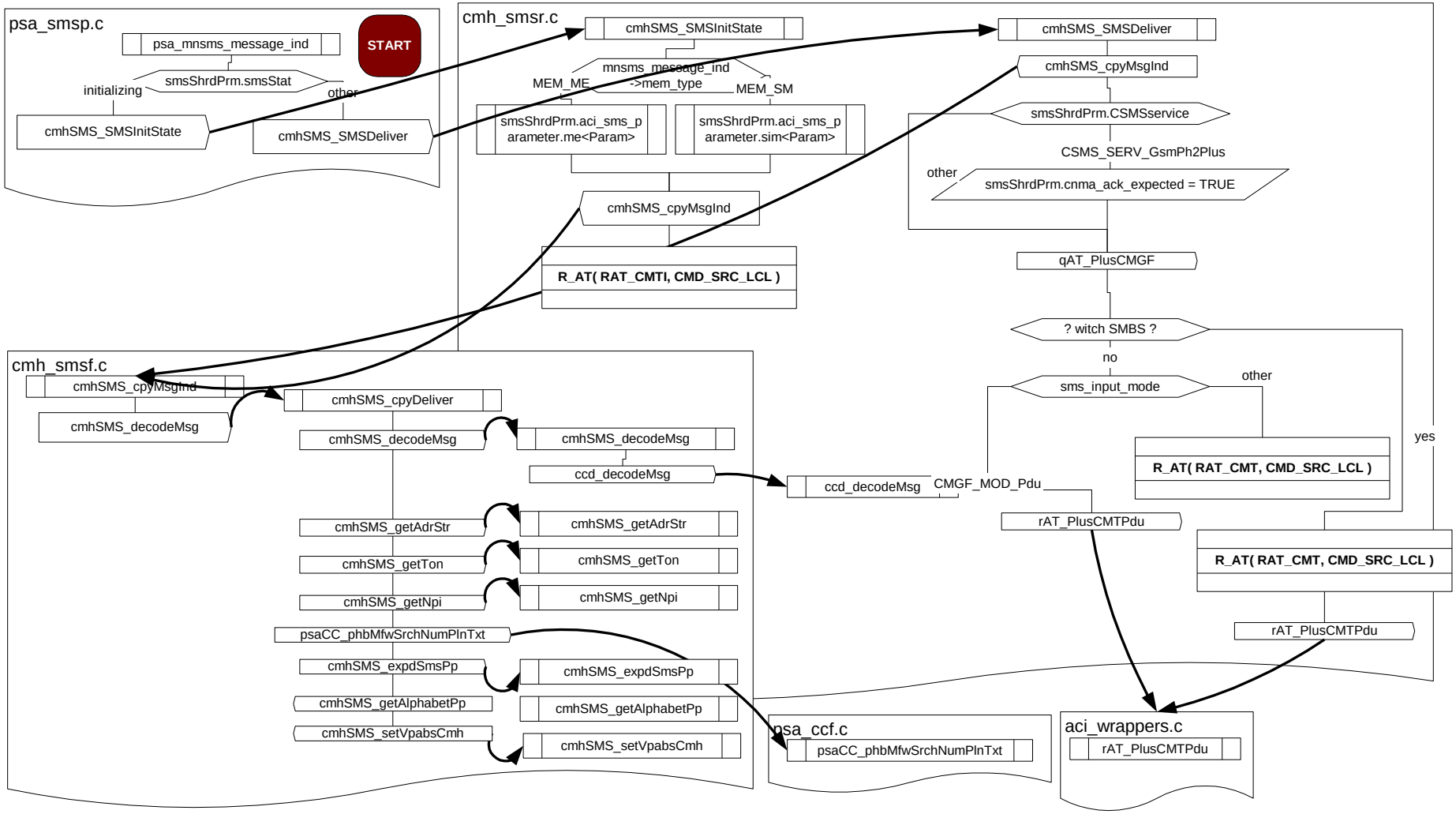
Now, in our picture below after the second relayed message, the application does not send the +CNMA because it is in state DATA-MODE or busy for other reasons. In this case a timer (15sec.) will be expired and all other remained messages will be stored into the non-volatile memory (in SIM or MS) with status "unread" and the +CNMI parameter <ds> and <mt> will be reset to '0'.

In the following scenario a content of three SMS in the buffer will be considered. The first SMS will be delivered successful. The second SMS cannot be received from the application (AT command interpreter) and has to be stored at the SIM or MS. Then, the third SMS has to be store into the non-volatile memory also.

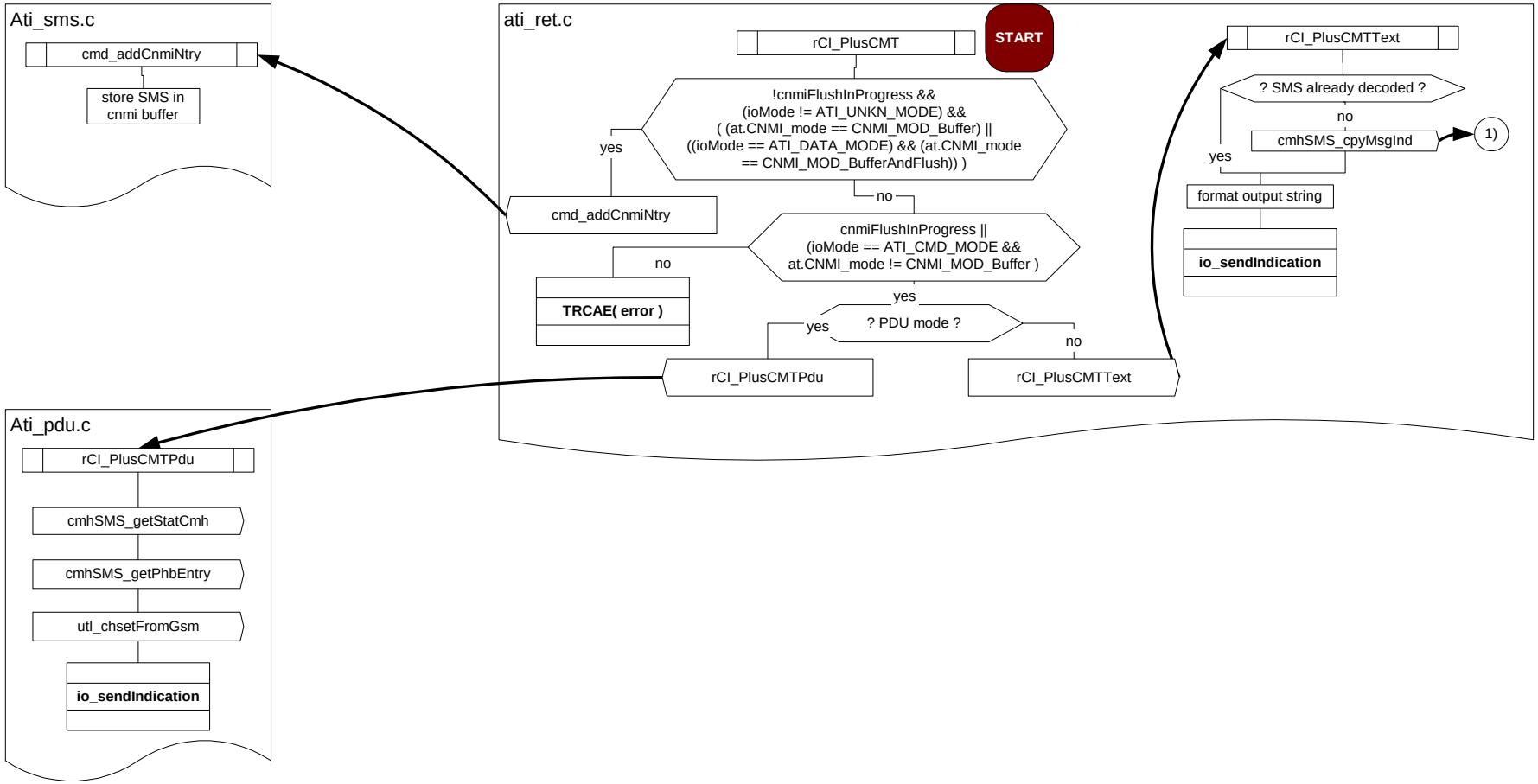

Timer
network
SMS
ACI
Application (Terminal)
Volatile memory (inside ACI)
atPlusCNMI() context
mnsms_configure_req
AT+CNMI=2,2,2,1,0
getMsg()
wait for acknowledgement
+CMT : <SMS_message>
start timer (15sec.)
atPlusCNMA() context
stop timer
AT+CNMA=[0|1]
uiMsgCount=getNbrOfEntrys()
? last Msg.?
no
wait for acknowledgement
getMsg()
+CMT : <SMS_message>
start timer (15sec.)
>15 sec
missing AT+CNMA
CNMI buffer
timer expired
non Volatile memory (SIM or device)
set <mt>=0, <ds>=0
mnsms_configure_req (<mt>=0, <ds>=0)
atPlusCMGW(*cl, srcId)
uiMsgCount=getNbrOfEntrys()
? last Msg.?
no
CNMI buffer
getMsg()
atPlusCMGW(*cl, srcId)
uiMsgCount=getNbrOfEntrys()
? last Msg.?
yes
timer context

4 Present implementation

4.1 Receiving an SMS

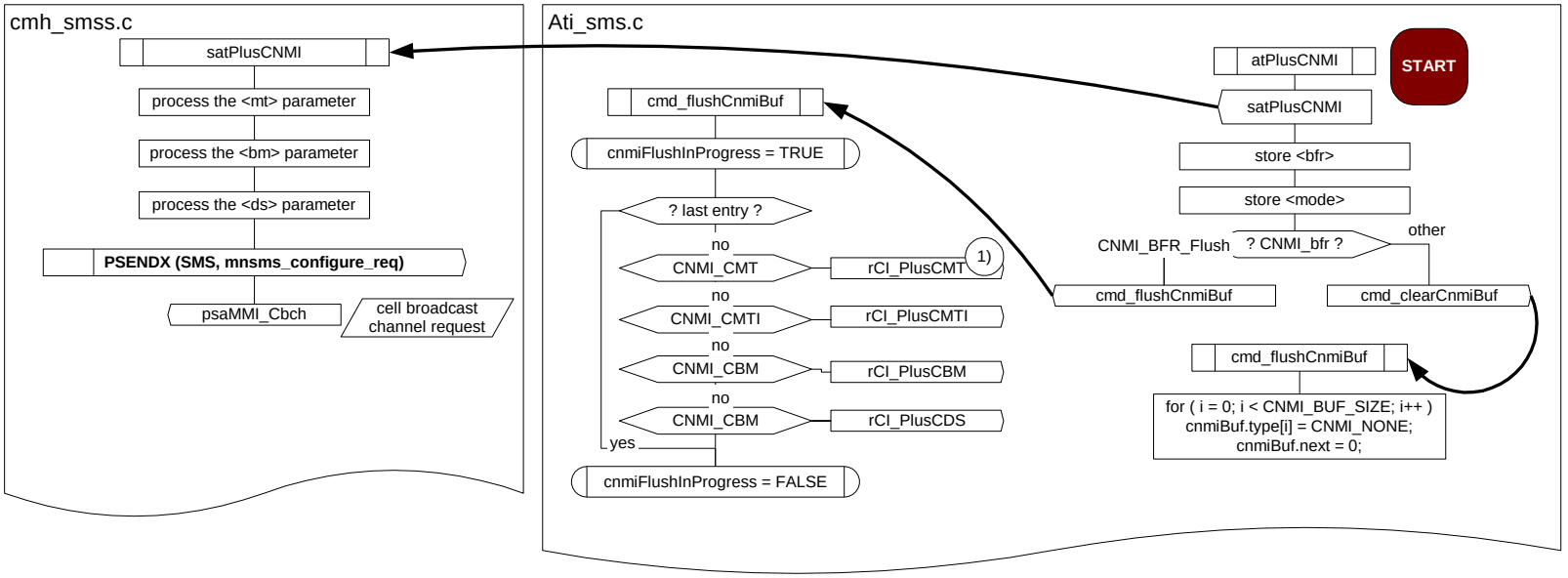


4.2 Execution of R_AT(RAT_CMT, …)



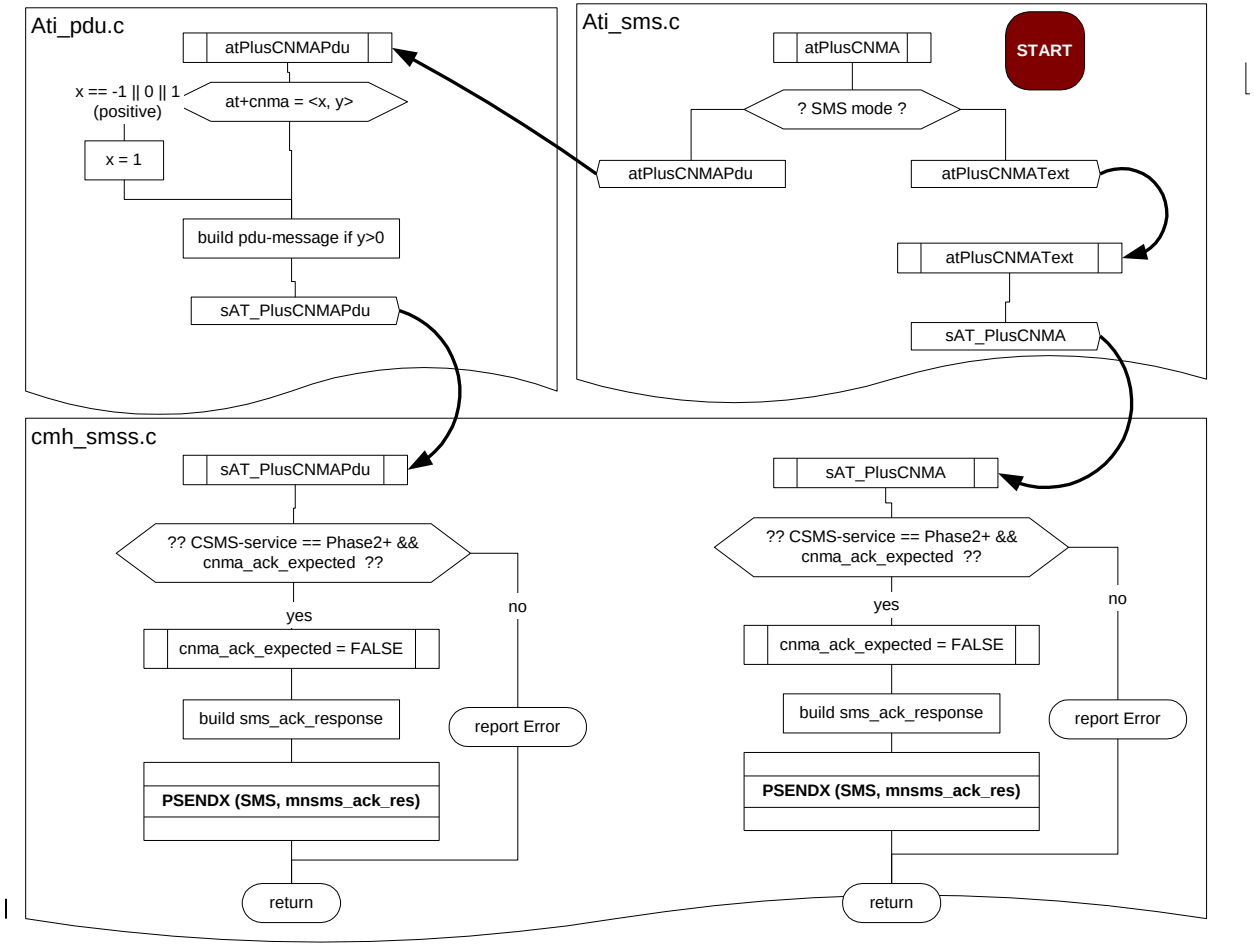
1) siehe 4.1

4.3 +CNMI handling (sAT_PlusCNMI)

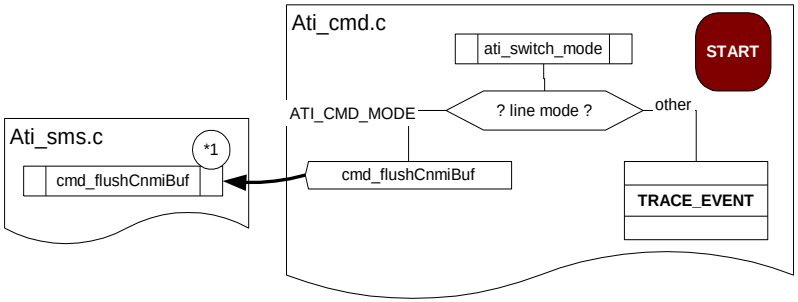


1) siehe 4.2

4.4 +CNMA handling

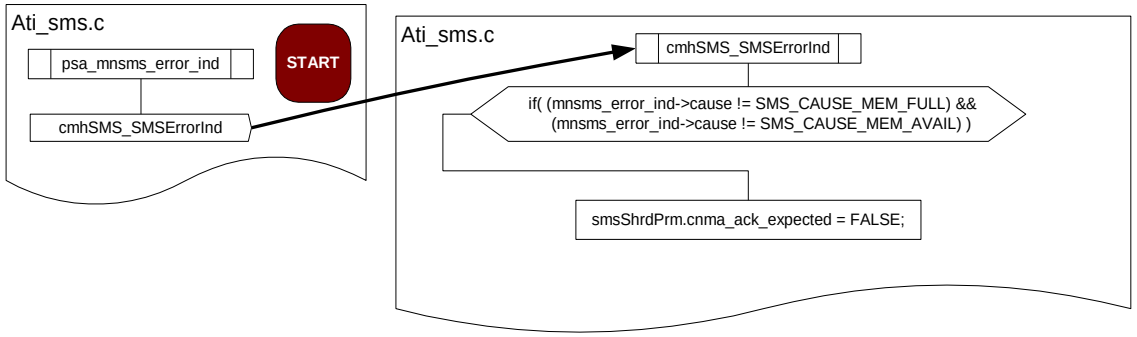


4.5 switch-mode handling



*1) see 4.3

4.6 Behavior in error case → psa_mnsms_error_ind() causing timeout in SMS



5 Changes in existing implementation

5.1 Changes in ACI routines

This chapter refers to pictures in chapter 4.1.

Additionally, the structure of CNMI buffer must be changed for better access by pointers and indexes.

5.1.1.1 New global variables

Adding of following global variables are urgent necessary:

- GLOBAL S16 waitForCnmaFromBuffer_SrcId = CMD_SRC_NONE;
  this variable indicates, that the ACI is waiting for an +CNMA acknowledge

5.1.1.2 New CNMI-buffer handling functions

- cmd_clearFirstMsgWithSrcIdInCnmiBuf(srcId); → clear the first entry in CNMI-buffer with 'srcId'
- cmd_flushCnmiBufOneByOne(srcId); → this is the entry function for flushing the CNMI-buffer in Phase2+ mode
- cmd_getNumberOfCnmiEntrys → get the number of entry's in the CNMI-buffer
- cmd_clearCnmiMessage(UINT16 uiIndex) → clears a entry in the CNMI-buffer on position uiIndex

5.1.2 founded errors while implementing

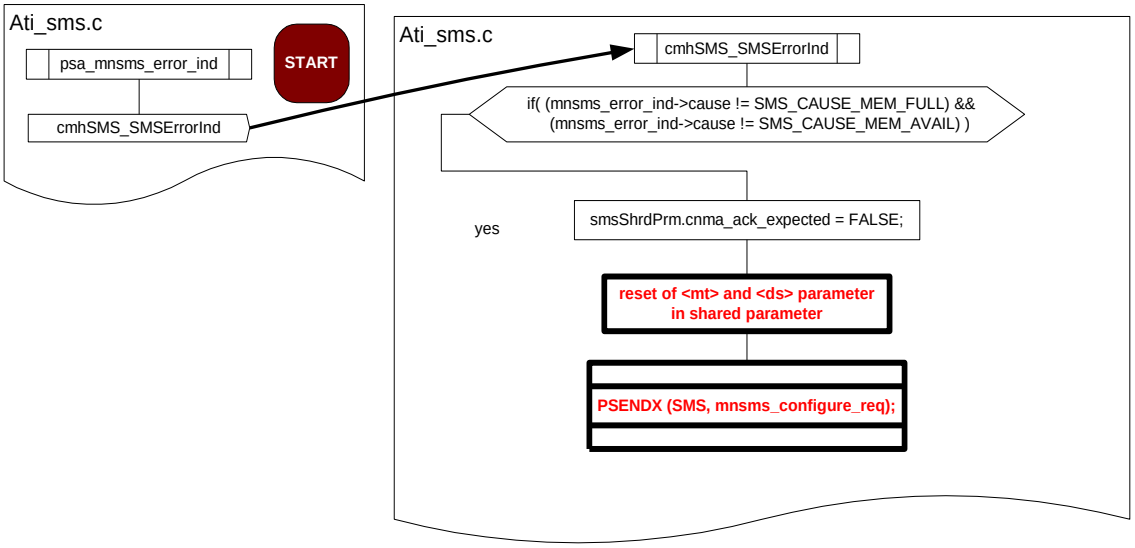
- clear p_sm buffer after each SMS output process (flushing CNMI buffer also)
- fix error at atPlusCNMA → non initialized 'ret' –value

5.1.3 Changes in cmhSMS_SMSErrorInd() [error handling from SMS entity]

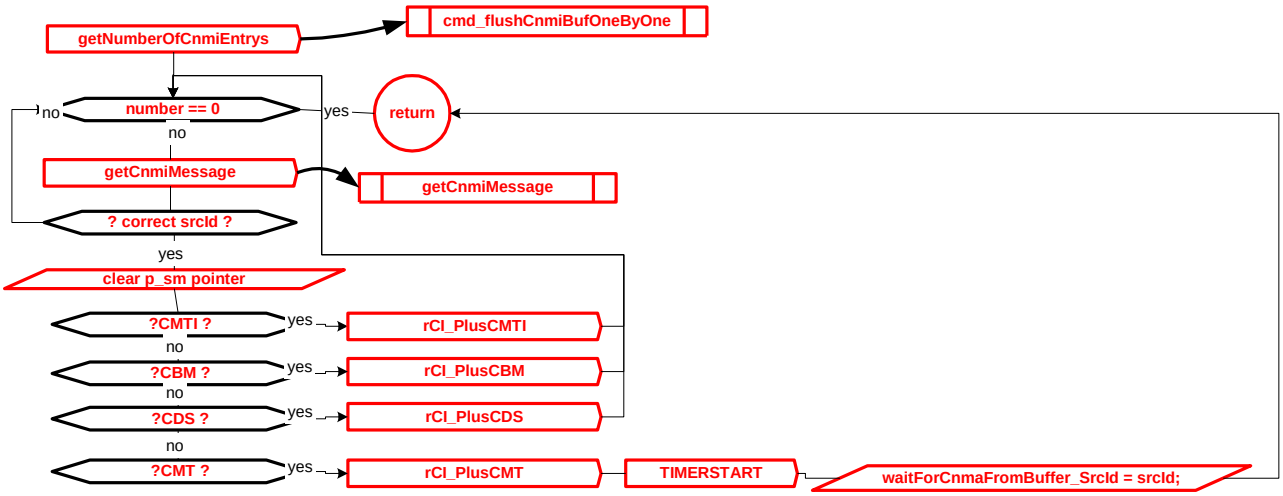
While the ACI is waiting for the AT+CNMA it can be possible that the SMS entity is sending a 'psa_mnsms_error_ind' that means a timer is expired because this AT+CNMA is missing. In this case the <mt> and <ds> parameter must be set to '0' and the SMS entity must be informed that these parameters has been changed. Following SMS will be buffered.

Changes must be made in cmhSMS_SMSErrorInd() at file ati_sms.c:

- reset the <mt> and <ds> parameter in shared memory parameter
- send a PSENDX (SMS, mnsms_configure_req) to SMS entity (see at sAT_PlusCNMI at file cmh_smss.c)



5.1.4 cmd_flushCnmiBufOneByOne(srcId);

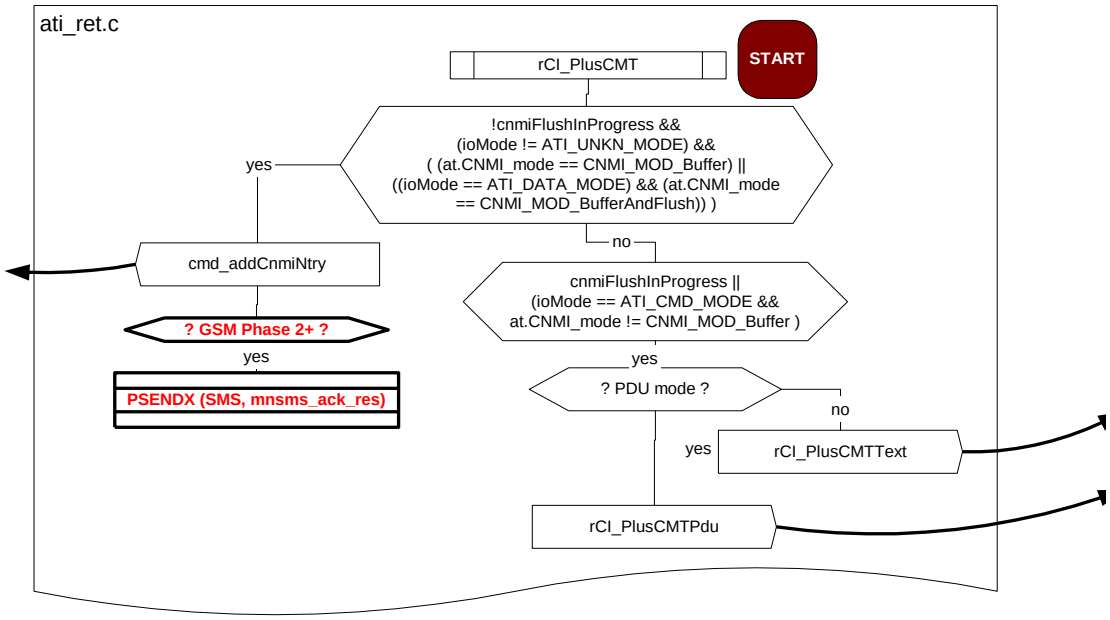


Changes in rCI_PlusCMT() [terminal output handling]

After calling this function, first a check whether the message has to be stored into the volatile memory shall be performed or not. It depends on the present IO mode and the settings in the CNMI mode parameter.

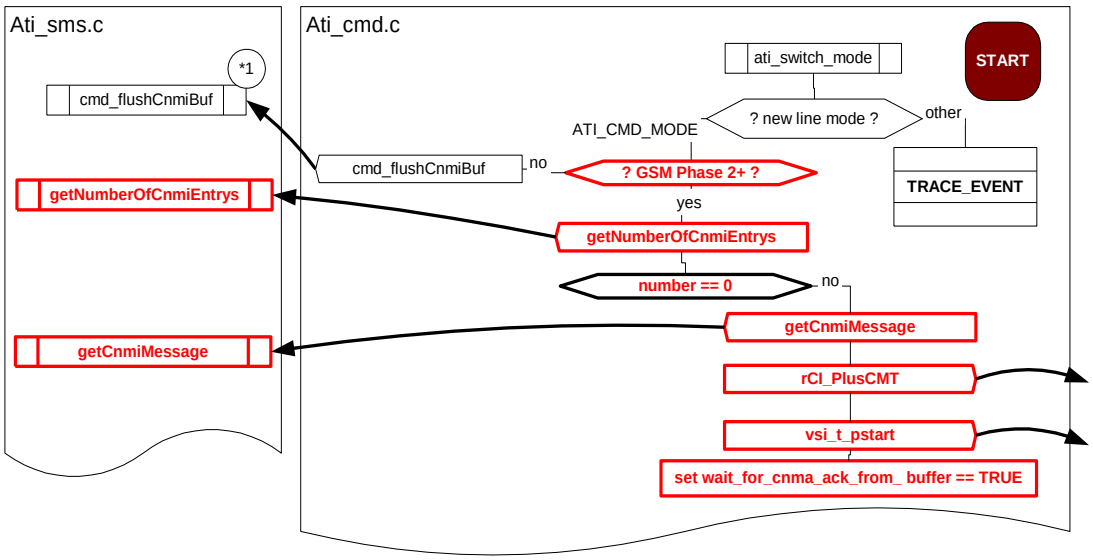
Changes must be made in rCIPlusCMT() at file psa_smsp.c:

- An acknowledge response shall be sent to the SMS entity after the SMS has been stored into the volatile buffer if the GSM Phase mode 2+ is enabled (AT+CSMS=1). Please refer to chapter '3.1.2.1 Acknowledgement of buffered received SMS' for the entire picture.



5.1.5 Changes in Ati_switch_mode() [UART-mode handling: data- to command mode]

If the UART-mode is switched from data- to command mode, first a check whether messages have to be received from the volatile memory shall be performed or not. This decision depends on the settings of the +CSMS command. If messages must be delivered to the application, a check into the volatile memory will follow, whether one or more SMS are inside. After the SMS has been send to the application a 15sec. timer will start and the ACI returns to the caller at least.

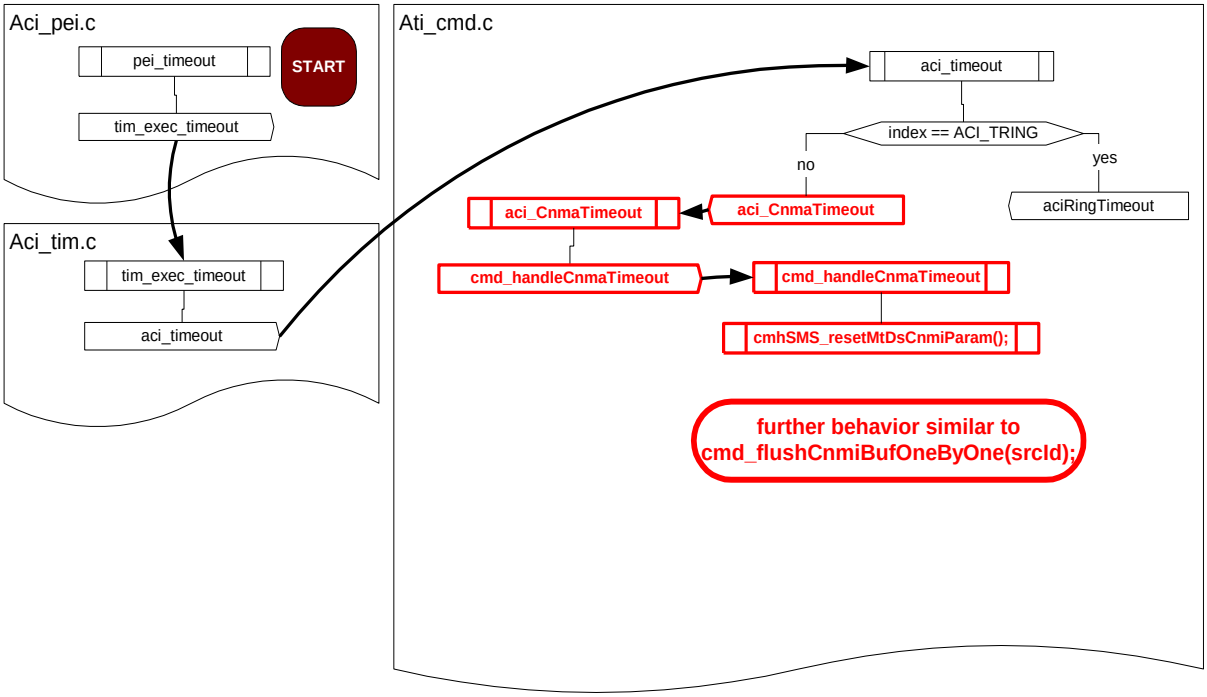


The timer will be start with the TIMERSTART macro. This function has following parameters:

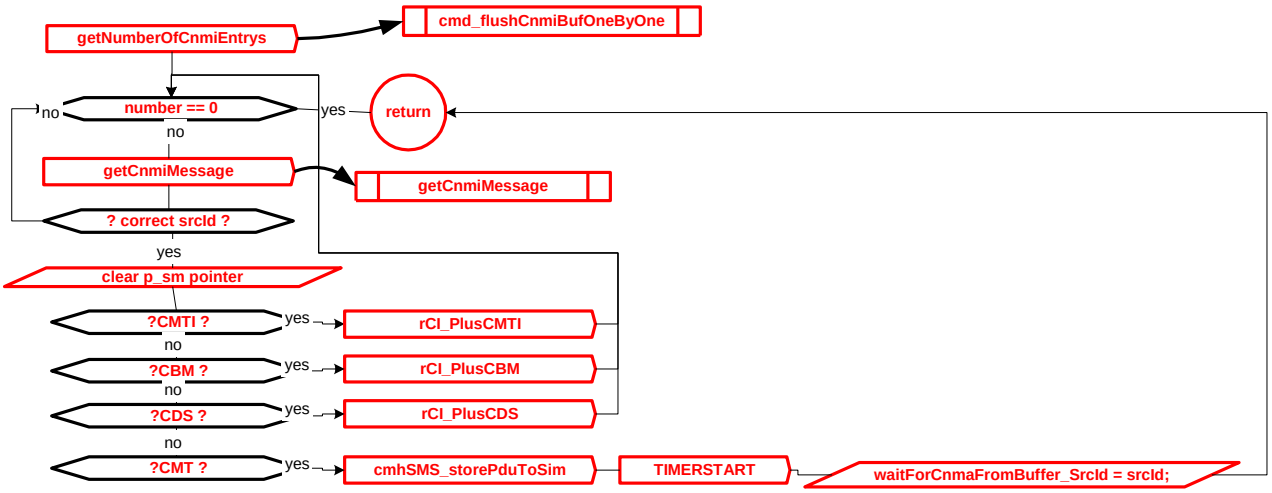
- CNMA timer handle → #define ACI_CNMA_TIMER_HANDLE to be defined in 'l4_tim.h'
- Timer value (in ms) → #define ACI_CNMA_TIMER_VALUE 15000 to be defined in 'ati_cmd.h'

5.1.6 Changes in aci_timeout() [timeout handling]

If the timer fails a primitive will income to 'pei_timeout()' function in 'aci_pei.c'. Further handling will done by 'tim_exec_timeout()'. Here the aci_timeout()' function will process further. This function will switch to 'aciCnmaTimeout()'. See details at following sheme:



5.1.6.1 Store PDU to non volatile memory function cmd_storeNextCnmiBufMsgToSim(void)



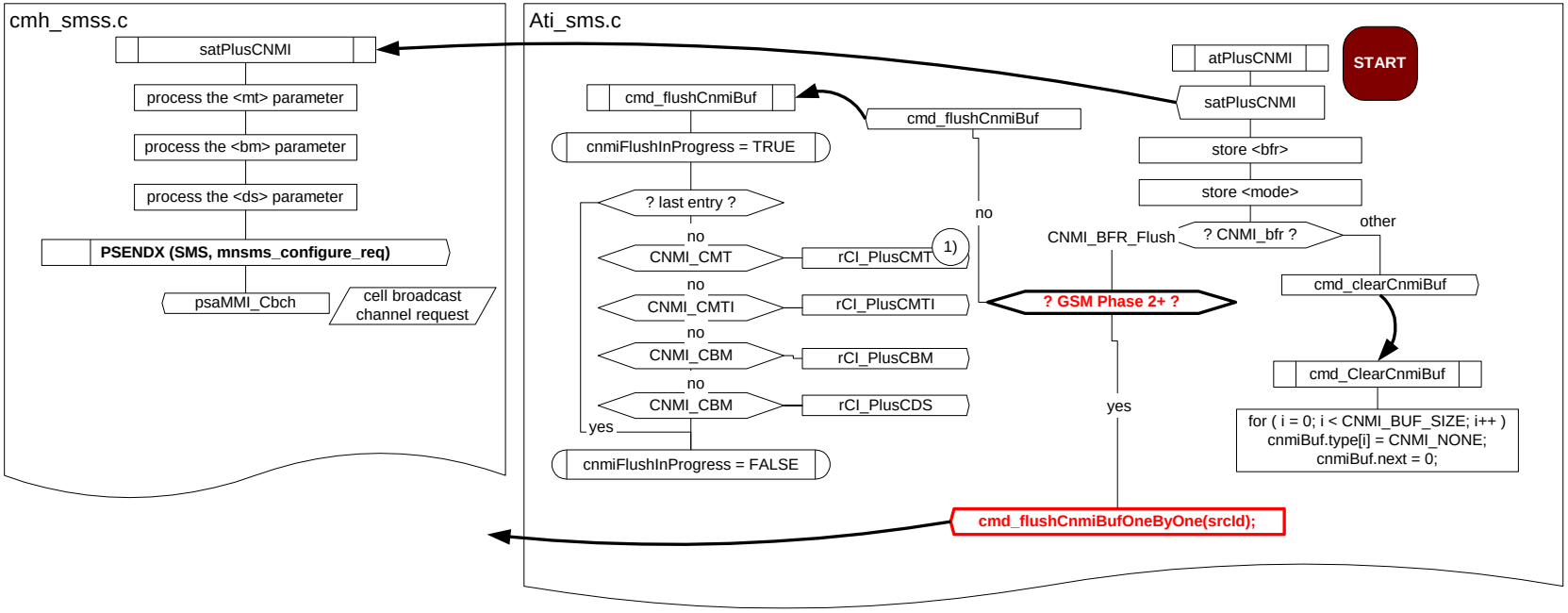
5.1.7 changes in cmhSMS_SMSStoCnf()

Because of storing an SMS onto the SIM or MS this function will called after the storing process is done. Because of possible storing of further messages these function must be changed.

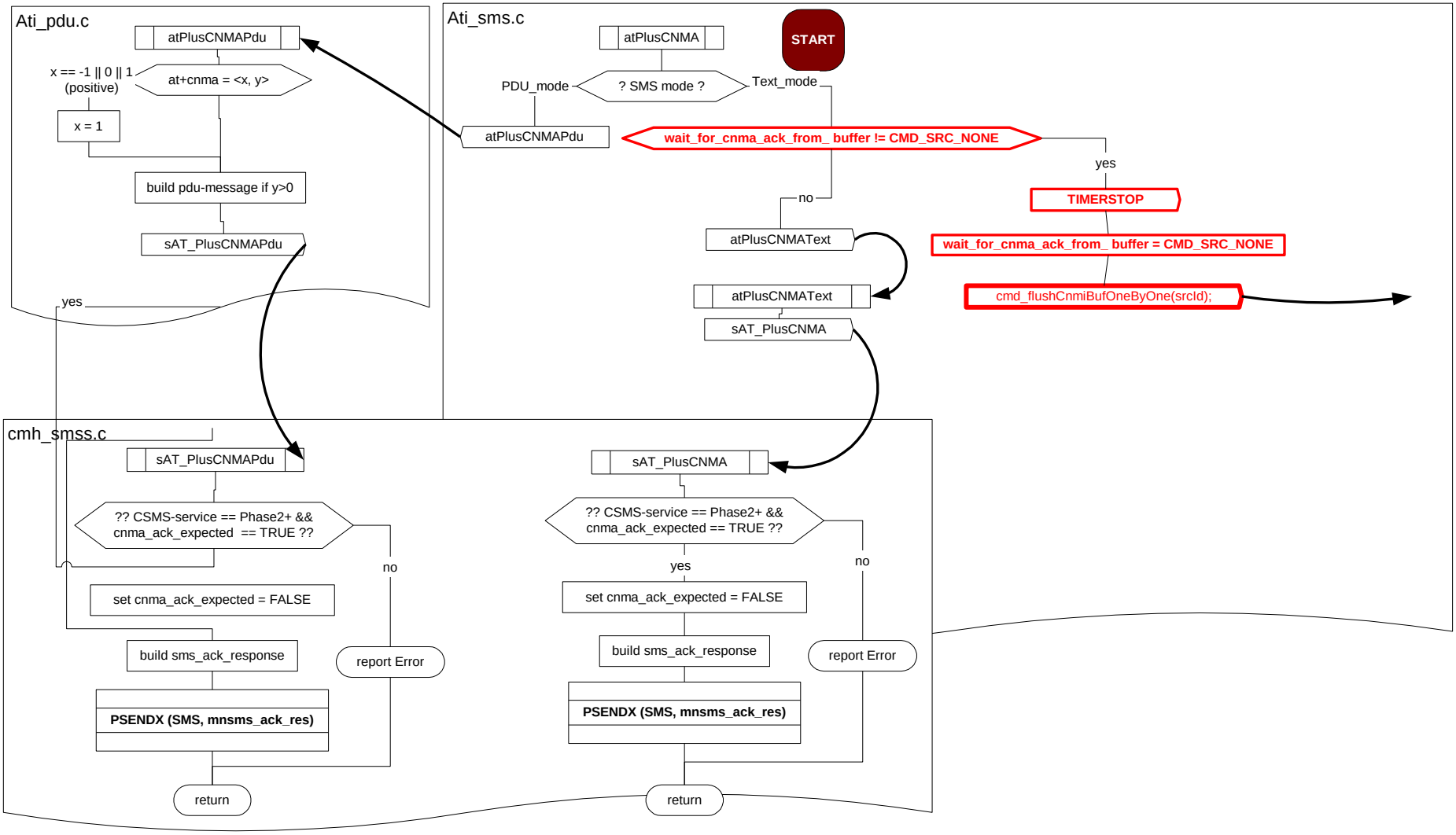
In this special case (`smsShrdPrm.uiInternalSmsStorage != CMD_SRC_NONE`) the recently stored message must be delete from CNMI-buffer (`md_clearFirstMsgWithSrcIdInCnmiBuf( smsShrdPrm.uiInternalSmsStorage );`). Then the next CNMI-buffer entry has to be stored on the SIM if existing.

5.1.8 Changes in atPlusCNMI()

If the user has entered the <bfr> command to flush the volatile buffer, a decision whether the SMS should be get one by one or all at once from that. If the buffer content should be get one by one, an additional function must be written. Then a +CMT message including the SMS will be generated. A timer will be started and the ACI is waiting for the AT+CNMA.



5.1.9 Changes in atPlusCNMA()



6 Test

The test of this feature comprises a simulation and a target test. Each kind of test has some benefits in contrast to the opposite kind of test. For example, you cannot test the behavior in error cases, because usually you are not to be able to provoke a network error.

6.1 Simulation test

At the simulation test all described scenarios as described in chapter 0 will be tested.

Following tests will be done:

6.1.1 Textmode

ASC256 : Test of more than one [3 times] incoming SMS (text mode) in Phase2+-mode

Test focus: ability to acknowledge of more than one incoming SMS by +CNMA

ASC257 : Test of more than one incoming SMS (text mode) in Phase2+-mode with timeout

Test focus: ability to process the 'psa_mnsms_error_ind()' at ACI. The 'mnsms_configure_req()' message should be send back to the SMS entity (Test Frame).

Test whether <mt> and <ds> is set to '0' and a +CNMA command is no more possible.

ASC258 : Test to acknowledgement of buffered SMS (text mode) in Phase2+-mode and flush

the CNMI-buffer including timeout test

Test focus:

- Test whether the 'mnsms_ack_res()' comes immediately after incoming the

'psa_mnsms_message_ind()'.

- Test whether the SMS has been stored into the CNMI buffer.

- Timeout test (remained messages in CNMI-buffer will store at the SIM/MS)

- Test to <mt> and <ds> is set to '0' and a +CNMA command is no more possible.

6.1.2 PDU mode

ASC441 : Test of more than one [3 times] incoming SMS (text mode) in Phase2+-mode

Test focus: ability to acknowledge of more than one incoming SMS by +CNMA

ASC442 : Test of more than one incoming SMS (text mode) in Phase2+-mode with timeout

Test focus: ability to process the 'psa_mnsms_error_ind()' at ACI. The 'mnsms_configure_req()' message should be send back to the SMS entity (Test Frame).

Test whether <mt> and <ds> is set to '0' and a +CNMA command is no more possible.

ASC443 : Test to acknowledgement of buffered SMS (text mode) in Phase2+-mode and flush

the CNMI-buffer including timeout test

Test focus:

- Test whether the 'mnsms_ack_res()' comes immediately after incoming the

'psa_mnsms_message_ind()'.

- Test whether the SMS has been stored into the CNMI buffer.

- Timeout test (remained messages in CNMI-buffer will store at the SIM/MS)

- Test to <mt> and <ds> is set to '0' and a +CNMA command is no more possible.

6.2 Target test

The target test will be done via a D-Sample and CRT. Following tests should be done:

- Normal acknowledge tests at PDU- and Text-mode
- Timeout scenarios