



Technical Document – Confidential

GSM PROTOCOL STACK

G23

SAT CLASS C + E

LL DESIGN SPECIFICATION ACI

Document Number:	8462.702.01.001
Version:	0.5
Status:	Draft
Approval Authority:	
Creation Date:	2002-Feb-01
Last changed:	2015-Mar-08 by XGUTTEFE
File Name:	SATClassC+E-ACI .doc

Important Notice

Texas Instruments Incorporated and/or its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products, software and services at any time and to discontinue any product, software or service without notice. Customers should obtain the latest relevant information during product design and before placing orders and should verify that such information is current and complete.

All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment. TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI products, software and/or services. To minimize the risks associated with customer products and applications, customers should provide adequate design, testing and operating safeguards.

Any access to and/or use of TI software described in this document is subject to Customers entering into formal license agreements and payment of associated license fees. TI software may solely be used and/or copied subject to and strictly in accordance with all the terms of such license agreements.

Customer acknowledges and agrees that TI products and/or software may be based on or implement industry recognized standards and that certain third parties may claim intellectual property rights therein. The supply of products and/or the licensing of software does not convey a license from TI to any third party intellectual property rights and TI expressly disclaims liability for infringement of third party intellectual property rights.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products, software or services are used.

Information published by TI regarding third-party products, software or services does not constitute a license from TI to use such products, software or services or a warranty, endorsement thereof or statement regarding their availability. Use of such information, products, software or services may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

No part of this document may be reproduced or transmitted in any form or by any means, electronically or mechanically, including photocopying and recording, for any purpose without the express written permission of TI.

Change History

Date	Changed by	Approved by	Version	Status	Notes
2002-Feb-01	RIE		0.1		1
2002-Mar-22	RM		0.2		2
2002-Jun-12	AK		0.3		3
2002-Sep-05	RM		0.4		4
2003-Jun-02	XGUTTEFE		0.5	Draft	

Notes:

1. Initial version

2. Update
3. Rework
4. Update and Number

Table of Contents

1.1	References	7
1.2	Abbreviations	7
1.3	Terms	7
2	Overview	7
3	Introduction	8
3.1	SAT Class C	8
3.2	SAT Class E	8
4	Software Environment	9
5	Limitations.....	12
5.1	SAT Class C	12
5.2	SAT Class E	12
6	Design.....	13
6.1	SAT Class C	13
6.2	SAT Class E	13
6.2.1	Basic Concept	13
6.2.2	Channel Establishment.....	13
6.2.3	Open Channel State Machine	16
6.2.4	Channel Table	16
6.2.5	Channel Release	17
6.2.6	Event Handling	18
7	Enhancements of SAT Message Catalogue	18
7.1	New Messages	18
7.2	New Structured Elements	19
7.3	New Basic Elements.....	19
8	Changes to existing Interfaces	20
8.1	ACI Functional Interface	20
8.2	AT Interface.....	21
9	Integration with Existing Software.....	22
9.1	Extension of Existing Data Types for SAT Class E.....	22
9.1.1	Module psa_cc.h.....	22
9.1.2	Module psa_sat.h.....	23
9.1.3	Module psa_sim.h	23
9.2	Modification of Existing Functions for SAT Class E	24
9.2.1	Module ati_sat.c.....	24
9.2.2	Module cmh_ccr.c	24
9.2.3	Module cmh_ccs.c	25
9.2.4	Module cmh_gmmr.c	25
9.2.5	Module cmh_l2rs.c	25
9.2.6	Module cmh_satf.c	26
9.2.7	Module cmh_satr.c	27
9.2.8	Module cmh_sats.c	27

9.2.9	Module cmh_smf.c	27
9.2.10	Module cmh_smr.c	27
9.2.11	Module gaci.c	27
9.2.12	Module gaci_srcc.c	27
9.2.13	Module dti_manager.c	28
9.2.14	Module psa_l2rs.c	28
9.2.15	Module psa_l2rp.c	28
9.2.16	Module psa_satf.c	28
9.2.17	Module psa_satp.c	28
9.2.18	Module psa_smf.c	29
9.2.19	Module psa_tras.c	29
9.2.20	Module psa_udpap.c	29
9.2.21	Module psa_udpas.c	29
9.2.22	Module smi_dmy.c	29
10	New Functions and Data Structures	29
10.1	Predefined Values	29
10.1.1	Module cmh_sat.h	29
10.1.2	Module psa_sat.h	29
10.1.3	Module psa_tcpip.h	29
10.2	New Data Types Introduced for SAT Class E	29
10.2.1	Module aci_cmh.h	29
10.2.2	Module psa_sat.h	30
10.2.3	Module psa_sim.h	32
10.3	New Functions for SAT Class E	32
10.3.1	Module ati_ret.c	32
10.3.2	Module cmh_satf.c	33
10.3.3	Module cmh_smf.c	40
10.3.4	Module dti_manager.c	40
10.3.5	Module psa_satf.c	41
10.3.6	Module psa_simp.c	42
10.3.7	Module psa_sims.c	42
10.3.8	Module psa_udpas.c	43
11	MSC	43
11.1	OPEN CHANNEL, confirmation phase	43
11.1.1	OPEN CHANNEL, command will be rejected because of unsupported parameters	43
11.1.2	OPEN CHANNEL, user rejects open channel establishment attempt	44
11.1.3	OPEN CHANNEL, user accepts open channel establishment attempt	44
11.1.4	OPEN CHANNEL, channel establishment allowed by Call Control by SIM	45
11.1.5	OPEN CHANNEL, channel establishment not allowed by Call Control by SIM	45
11.2	OPEN CHANNEL, establishment phase	46
11.2.1	OPEN CHANNEL, immediate channel establishment, CSD, UDP	46
11.2.2	OPEN CHANNEL, immediate channel establishment, CSD, non-transparent	47
11.2.3	OPEN CHANNEL, immediate channel establishment, CSD, transparent	48
11.2.4	OPEN CHANNEL, immediate channel establishment, CSD, rejected by the network ..	49
11.2.5	OPEN CHANNEL, immediate channel establishment, CSD, failure within the data chain 50	
11.2.6	OPEN CHANNEL, immediate channel establishment, GPRS, UDP	51
11.2.7	OPEN CHANNEL, immediate channel establishment, GPRS, SNDCCP	52
11.2.8	OPEN CHANNEL, immediate channel establishment, GPRS, rejected by the network	53
11.2.9	OPEN CHANNEL, immediate channel establishment, SIM connection failure	53
11.2.10	OPEN CHANNEL, on demand channel establishment	54
11.3	SEND DATA	54
11.3.1	SEND DATA, on demand	54
11.3.2	SEND DATA, immediate, CSD, UDP	55

11.3.3	SEND DATA, immediate, GPRS, UDP	56
11.3.4	SEND DATA, immediate, CSD, rejected by the network	57
11.3.5	SEND DATA, immediate, CSD, failure within the data chain	58
11.3.6	SEND DATA, immediate, GPRS, rejected by the network	59
11.4	RECEIVE DATA	60
11.4.1	RECEIVE DATA	60
11.5	CLOSE CHANNEL	60
11.5.1	CLOSE CHANNEL, CSD, UDP	60
11.5.2	CLOSE CHANNEL, CSD, non-transparent	61
11.5.3	CLOSE CHANNEL, CSD, transparent	62
11.5.4	CLOSE CHANNEL, GPRS, UDP	63
11.5.5	CLOSE CHANNEL, GPRS, SNDCCP	63
11.6	GET CHANNEL STATUS	64
11.6.1	GET CHANNEL STATUS	64
11.7	CHANNEL STATUS/DATA AVAILABLE EVENT	64
11.7.1	Event Download	64
11.7.2	CHANNEL STATUS EVENT	65
Appendices		66
A.	Acronyms	66
B.	Glossary	66

List of Figures and Tables

List of References

- [ISO 9000:2000] International Organization for Standardization. Quality management systems - Fundamentals and vocabulary. December 2000

1.1 References

- [3GPP 11.14] 3GPP TS 11.14 version 8.9.0 Release 1999, Specification of the SIM Application Toolkit for the Subscriber Identity Module - Mobile Equipment (SIM-ME) interface
- [1] 8411.802.00.110, ACI - Application Control Interface, Functional Interface Description
- [2] 6147.609.99.008, SAT - Air Interface Message Specification

1.2 Abbreviations

ACI	Application Control Interface (AT Commands)
CCD	Condat Coder Decoder
CSD	Circuit Switched Data
GPRS	General Packet Radio Service
IP	Internet Protocol
SAT, STK	SIM Application Toolkit
SNDCP	Sub-Network Dependent Convergence Protocol
SIM	Subscriber Identity Module
PPP	Point to Point Protocol
L2R	Layer 2 Relay
TCP	Transport Control Protocol
UDP	User Datagram Protocol
URL	Uniform Resource Locator

1.3 Terms

2 Overview

This document describes the design and implementation of SAT letter class c and e, with respect to the ACI software module. The document is structured as follows:

Introduction:	A short summary of the functionality behind SAT letter class c and e. This will be from a high level point of view to provide a common understanding. Details will be discussed in the Design chapter.
Software Environment	This chapter will provide an overview of the existing ACI module environment. It will help to understand the impact of SAT letter class c and e on the G23 protocol stack.
Limitations:	Not all of the functionality described in [3GPP 11.14] for SAT letter class c and e could be implemented at this stage of development. This section lists all limitations which are present.
Design:	This is the main chapter of the document and will show the details of the design for SAT letter class c and e. Due to the fact that class c and e do not have much in common, this chapter is separated for the different classes.

Enhancements of SAT Message Catalogue:	The SAT implementation in ACI uses the CCD for coding and decoding of SAT commands and results in general. This chapter will give an overview about the enhancements for the SAT message catalogue for SAT letter class c and e.
Changes to Existing Interfaces:	This chapter lists the changes to the functional and AT interface of ACI with respect to SAT letter class c and e.
Integration with Existing Software:	This chapter will describe how the extension of SAT letter class c and will fit into the existing software of ACI.
New Functions and Data Structures:	All functions and data structures which have to be developed for the implementation of SAT letter class c and e are listed here.
MSC:	A list of MSCs is given in this chapter to describe the dynamic behavior of ACI with its surrounding software entities.

3 Introduction

It is assumed that the reader has a basic understanding about SIM application toolkit mechanisms and functionalities. This document will focus only on SAT class c and e features and functionality. However, sometimes other SAT features have to be taken into account. These are not explained in detail, but could be found in [3GPP 11.14].

The SIM Application Toolkit provides mechanisms, which allow applications, existing in the SIM, to interact and operate with any ME that supports the specific mechanism(s) required by the application.

The following will give a rough overview about what SAT letter class c and e stand for and how it should work within a ME. The introduction will focus on the basic functionality of these features. More details could be found later on in the Design chapter.

3.1 SAT Class C

SAT class c deals with browser launch and termination. A new command is introduced which indicates to the ME that the proactive SIM will request to launch a browser. If the ME supports this class c command, it will indicate this by setting the LAUNCH BROWSER bit during profile download to the SIM.

The LAUNCH BROWSER command contains at least a URL to connect to and some optional additional parameters which qualify the type of connection or browser to be used. When the ME receives a LAUNCH BROWSER command it has to check whether the command could be executed or if conditions apply which prevents the ME from command execution. In this case the ME will terminate the command immediately. A cause will be given to SAT describing the problem.

If the ME has accepted the LAUNCH BROWSER command it has to set up a connection, start the browser application and load the contents provided at the requested URL. Once a final result of that operation is available, the LAUNCH BROWSER command will be terminated and a respective result has to be sent to SAT.

In addition to the LAUNCH BROWSER command the BROWSER TERMINATION EVENT is also part of SAT class c. The proactive SIM can request to be informed about browser termination by setting the BROWSER TERMINATION EVENT in the SET UP EVENT LIST command. If the ME accepts the event, every browser termination has to be indicated to SAT using the common event download mechanism.

3.2 SAT Class E

SAT Class e stands for the bearer independent protocol support. Therefore a set of new commands (OPEN CHANNEL, CLOSE CHANNEL, SEND DATA, RECEIVE DATA and GET CHANNEL STATUS) and two events (DATA AVAILABLE and CHANNEL STATUS) has been introduced to the specification. A proactive SIM card, which supports class e is able to establish a data channel with the ME, and through the ME to a remote Server in the Network. The SIM provides information for the ME to select an available bearer at the time of channel establishment. The ME then allows the SIM and the Server to exchange data on this channel, transparently.

If the ME supports class e commands it has to indicate this by setting the appropriate bits in the terminal profile. The SIM card sends a OPEN CHANNEL command to ME in order establish a data connection with a remote server. Basically the SIM card has to decide what kind of bearer it wants to use, CSD or GPRS. The OPEN CHANNEL command holds the necessary

parameters for the selected bearer and for the mode of channel establishment. SAT class e defines two modes for the channel establishment, immediate and on demand. In case of an immediate channel, the ME has to set up the data channel with the trigger of the OPEN CHANNEL command. For an on demand channel, the OPEN CHANNEL command transports only the channel parameters. ME has to wait with the channel establishment until it receives a SEND DATA command from SAT, which indicates an immediate channel establishment.

After the reception of an OPEN CHANNEL command, ME has to check the parameters whether it is able to perform the command or not. If parameters of the OPEN CHANNEL command prevent the ME from establishing such a data channel (e.g. parameters are not supported or missing) the command will be terminated immediately and a meaningful result code is sent to SAT. If Call Control by SIM is allocated and activated for the SIM card, CSD data channels have to be checked by SIM prior to channel establishment. If the SIM card rejects its own channel establishment, the OPEN CHANNEL command will be terminated immediately with the appropriate result code sent to SIM.

Once the verification of the OPEN CHANNEL command has been performed successfully, the user will be informed about the pending data channel establishment. If the user rejects the data channel, the command will be terminated immediately and the reason indicated to SAT. In case that the user accepts the data channel it depends on the establishment mode of the OPEN CHANNEL command how the ME will proceed. For an on demand channel the OPEN CHANNEL command will be terminated successfully and the ME will wait for the first SEND DATA command which indicates an immediate channel establishment. If the channel mode was immediate, the ME will start to set up the requested channel. The final result of the establishment attempt will terminate the OPEN CHANNEL command and inform SAT about the channel status.

If a channel has been established by SAT, the SEND DATA and RECEIVE DATA commands will be used to exchange data with the remote server. The ME has to provide a data buffer for temporary storage of channel data to support the asynchronous character of the SAT data channel. The proactive SIM can request to be informed about a channel status change and/or available data by setting the CHANNEL STATUS EVENT and/or DATA AVAILABLE EVENT in the SET UP EVENT LIST command. If the ME accepts these events, a drop of the channel and/or received data has to be indicated to SAT using the common event download mechanism.

If the SIM wants to release the data channel, it sends the CLOSE CHANNEL command to ME. Upon receiving this command, the ME stops data transmission for that channel and releases the channel. Data stored in the temporary buffer will be discarded. ME confirms channel disconnection by a successful command termination.

The GET CHANNEL STATUS command could be used by SAT to ask for the current status of all available ME channels. The status will be forwarded to SAT during command termination.

4 Software Environment

The following figure shows the software environment of ACI with the relevant modules and entities regarding the design for SAT Class c and e. Other entities and modules of the G23 protocol stack have been suppressed in the figure because they are of no relevance.

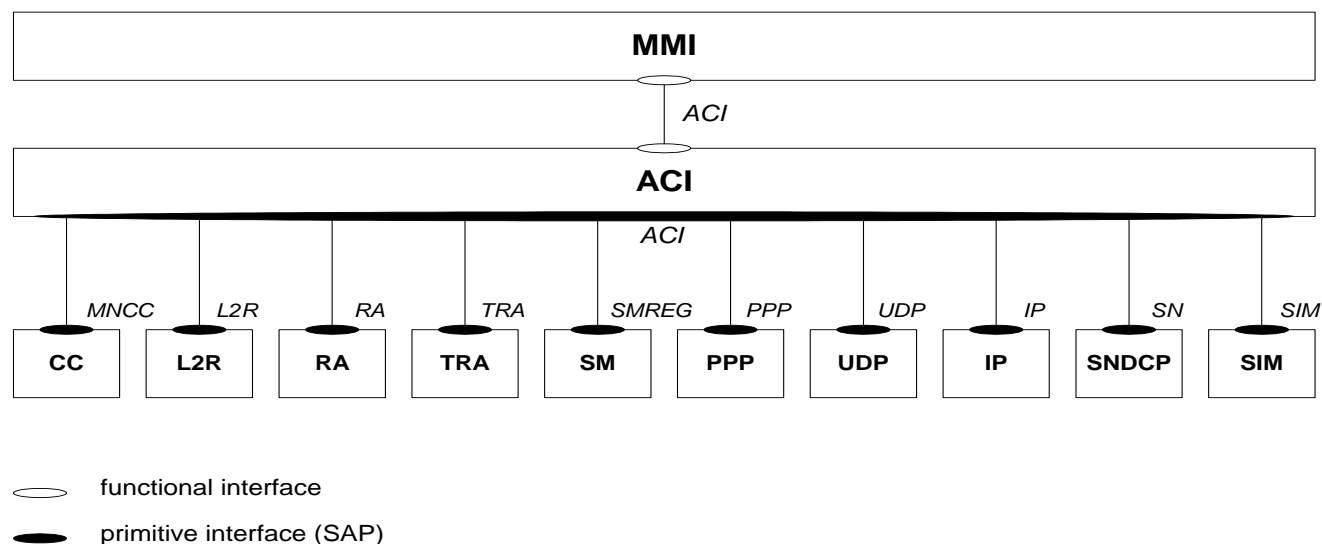


Figure 1: ACI Software Environment

A short summary of the interfaces shown in figure 1 should help to understand for what reason the entities and modules are used to implement the SAT class c and e features. These interfaces are mainly used to control the different kind of data channels requested by SAT.

Functional Interface ACI:

The functional interface of ACI (including the AT interface to external users) provides a subset of functions to handle SAT commands. On one hand it is used to pass commands and responses between MMI and SIM for SAT commands, which are not handled by ACI (e.g. display and menu commands). On the other hand it is used to inform the MMI about SAT commands and responses handled by ACI, and to alert the user about pending open channel requests from SAT. In the latter case the user uses the standard AT functions for ATA, ATH or AT+CHUP to accept or reject the channel establishment.

MNCC SAP:

For SAT Class e the MNCC SAP is used to establish, release and maintain every kind of CSD channel requested by SAT. The circuit switched connection serves as a basis for the different data channel configuration (transparent, non-transparent or UDP).

L2R SAP:

The L2R SAP is used to activate or deactivate the L2R entity in case of non-transparent or UDP CSD channels.

RA SAP:

The RA entity is a service layer for CSD channels. The RA SAP will be used to activate or deactivate the RA entity in case SAT has requested a CSD channel.

TRA SAP:

The TRA SAP is used to activate or deactivate the TRA entity in case of transparent CSD channels.

SMREG SAP:

In case of a GPRS channel ACI has to set up a PDP context within the GPRS network. The SMREG SAP is used to activate or deactivate the PDP context requested by SAT.

PPP SAP:

For a UDP CSD channel PPP is needed in the chain of data entities. This SAP is used to activate or deactivate the PPP entity.

UDPA SAP:

The UDPA SAP is used to activate or deactivate the UDP in case of a UDP CSD or GPRS channel requested by SAT.

IPA SAP:

The IP entity is a service layer for UDP channels for CSD or GPRS. The IPA SAP is used to activate, configure or deactivate the IP entity.

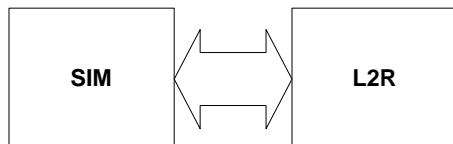
SN SAP:

The SN entity is used to switch the SNDCP entity from a null device connection to the SIM entity after the PDP context has been successfully activated. This is only necessary in case of a plain GPRS data channel (e.g. no UDP connection requested by SAT).

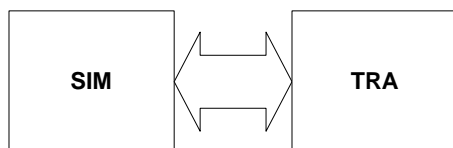
SIM SAP:

The SIM SAP is the main SAP used for implementing SAT class c and e. The SIM entity indicates all SAT commands, which are not handled by SIM to ACI. ACI will use the SIM SAP to send commands and responses to SAT. In addition to that the SIM SAP serves also as the interface to configure the SIM data connection for the requested data channel.

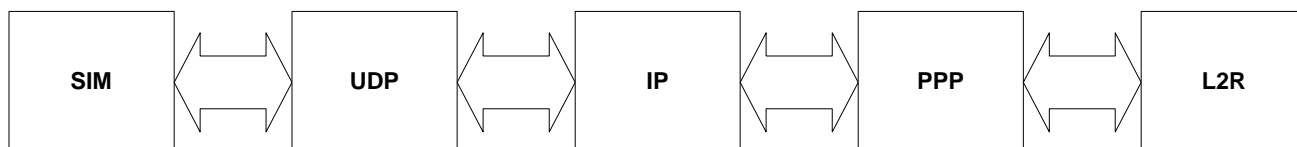
Besides the interfaces mentioned above to control the establishment and release of SAT data channels, the DTI manager is used to connect the different data entities for the desired channel configurations. The following figure shows the supported data chains for CSD channels.



SAT non-transparent data channel configuration
(This configuration will be the default, if a default CSD channel is requested by SAT)



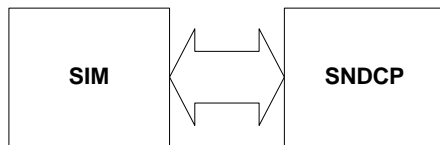
SAT transparent data channel configuration



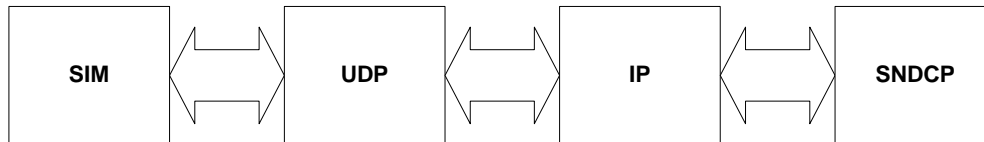
SAT UDP data channel configuration

Figure 2: Possible Data Chains for CSD channels

For GPRS data channels the possible configurations are shown in the next figure.



SAT packet data channel configuration
(This configuration will be the default, if a default GPRS channel is requested by SAT)



SAT UDP data channel configuration

Figure 3: Possible Data Chains for GPRS channels

5 Limitations

This section shows the limitations for the SAT class c and e design from the ACI point of view. These limitations are due to the current implementation of ACI or other entities of the G23 protocol stack. That does not mean that the implementation of these features is impossible in general, but for the current stage of the protocol stack development.

5.1 SAT Class C

No limitations

5.2 SAT Class E

- TCP as interface transport level is not supported and will be rejected by ME
- IP addresses of version 6 are not supported and will be rejected by ME
- If the UDP instance is already in use, a channel request for UDP will be rejected
- Only one bearer independent protocol channel is supported by ME
- In case of a CSD channel request, bearer capability parameters will not be forwarded to Call Control by SIM. Also bearer modifications by SIM will not be detected by ME
- In case that Call Control by SIM modifies a CSD channel into a SS or USSD transaction, the OPEN CHANNEL command will be treated as failed and the command will be terminated.
- The redial mechanism for CSD channels is only partly supported in ACI. Redial parameters will be indicated to MMI and the call parameters are stored for several additional attempts. It is the task of the MMI to force the redial attempt or terminate the command if redial conditions exceeds.
- In case that Call Control by SIM modifies a CSD channel into a SS or USSD transaction, the OPEN CHANNEL command will be treated as failed and the command will be terminated.

6 Design

6.1 SAT Class C

6.2 SAT Class E

6.2.1 Basic Concept

With SAT class e the proactive SIM card is able to establish a data channel with a remote server in the network. This feature can be separated into two main tasks. One task deals with the establishment, maintenance and release of the channel and the other task is responsible for the data transmission between the proactive SIM card and the remote server. The basic concept of the implementation for SAT class e will follow this separation. The design for SAT class e is a split design between the two entities ACI and SIM. ACI is responsible for channel management, where SIM takes over the task of data transmission between SAT and the data channel. The following figure shows the concept schematically.

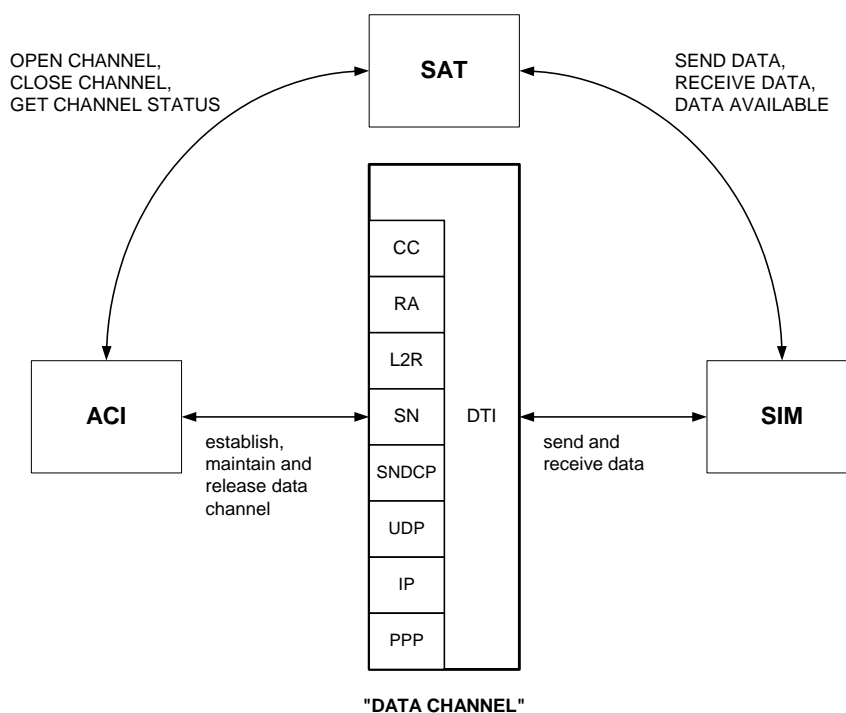


Figure 4: Separation between ACI and SIM

A data channel could be characterized as a network connection using a certain bearer service, several data entities that are in charge to support a specific data protocol and a couple of DTI connections which are necessary to glue the components together and to insure the flow of data. As the above figure shows ACI will handle SAT commands, which are used to open and close a channel and to provide channel status information. The SIM entity instead is responsible for the data plane to send, receive and indicate available data. To establish or release a certain data channel, ACI accesses the entity control ports and the DTI manager. The SIM entity will use the DTI connections to exchange data between SAT and the data channel. The following description will focus on the ACI part of the implementation only. The SIM part could be found in a separate low-level design document.

6.2.2 Channel Establishment

The SAT request to open a data channel can be splitted into two phases, the confirmation phase and the establishment phase. The following figure shows the principle of command execution and the separation into these phases.

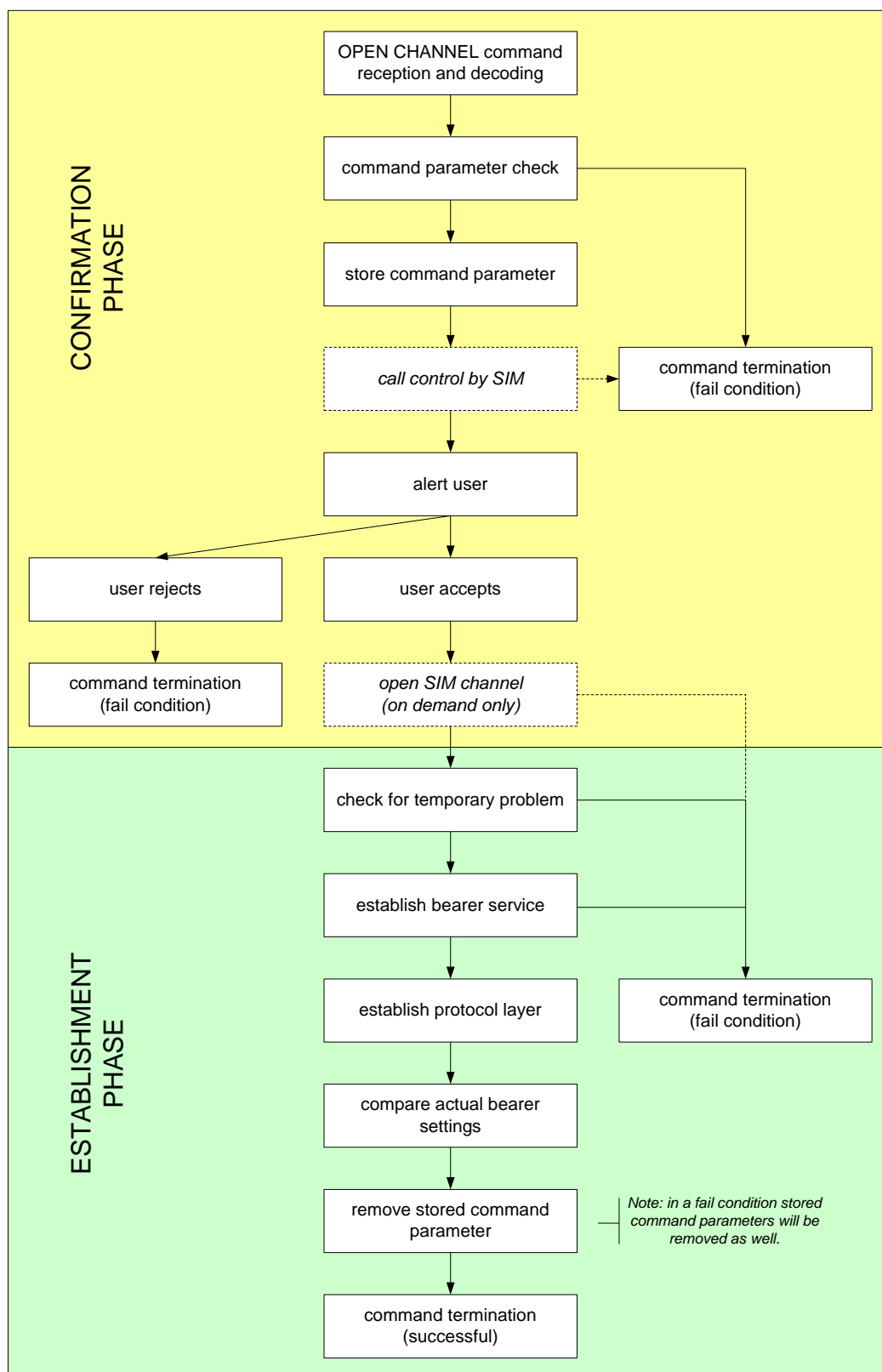


Figure 5: Principle of Channel Establishment

Confirmation Phase:

After reception of the OPEN CHANNEL command, ACI enters the Confirmation Phase. CCD will be used for further decoding of the OPEN CHANNEL command. All command parameters will be identified and a basic verification of the command take place. The command will be checked for channel requirements that cannot be fulfilled due to missing ME capabilities

(e.g. GPRS or CSD is requested but is not part of ME, IP address format not supported, buffer size too big, etc.). In case the verification fails, the command will be terminated immediately with an appropriate result cause.

Otherwise the command parameters will be stored for later use during the Establishment Phase. For CSD channel requests the bearer service parameters (called party number, data rate, etc.) are stored in a call table entry. For GPRS channels the quality of service parameters are stored by defining a PDP context. For both types of channels, additional parameters, which are important for the protocol layers are stored separately in dynamically allocated memory.

The next step of command execution is only valid for CSD channel requests (indicated by the dashed line in figure 5). If the SIM service *Call Control by SIM* is activated and allocated in the SIM service table, the bearer service parameters have to be passed to SAT for verification purposes. Command execution will be paused in this case until a reply from SAT is available. In case SAT will not allow to set up the channel or modifies the bearer type so that no data channel could be established, the command will be terminated with an appropriate result cause. If the channel set up is granted by SAT, then command execution will proceed as normal.

After all these preceding steps the user will be informed about a pending SAT data channel request (unsolicited AT result %SATA:). It is now the choice of the user to accept or reject the SAT data channel. In conformance with the handling for the SAT command SET UP CALL, the same AT commands are used for a user reply. The AT command A will accept the channel, whereas the commands H or +CHUP will reject the channel. In the latter case the command will be terminated with the appropriate result cause and stored parameters are removed. If the user accepts the channel it depends on the type of channel establishment what the next step will be.

In case of on demand channel establishment the channel set up will be postponed to a later time and explicitly indicated by SAT. Therefore only a SIM channel will be opened and the command will be terminated (indicated by the dashed line in figure 5). If immediate channel establishment is requested by SAT, the Confirmation Phase will be left and ACI enters the Establishment Phase.

Establishment Phase:

ACI will enter the Establishment Phase after one of two conditions is reached. The first condition is fulfilled after the Confirmation Phase has ended for an immediate channel request (see above). A predecessor to reach the second condition is that an OPEN CHANNEL command with the establishment mode set to on demand, has been performed successfully. After that ACI will enter the Establishment Phase as soon as the first SEND DATA command has been received, which indicates an immediate channel establishment.

Before the channel establishment will be started, an additional check for temporary problems is performed (e.g. ME is currently busy on a call or SS transaction, or the GPRS mobile class does not permit the channel establishment). In such a case command execution will be aborted and the command will be terminated with an appropriate result cause.

Otherwise ACI will start to set up the desired channel with the command parameters that have been stored from the OPEN CHANNEL command. After a network connection has been established for the requested bearer service, the data protocol entities will be installed. If one of the participating entities indicates a fail condition, the whole connection will be released and the command is terminated with a result indicating the reason. If the establishment of the channel was successful, the SIM entity will be finally connected with the data channel.

The actual bearer service parameters are compared with the requested ones. SAT will be informed about the current settings, when the command will be terminated successfully. Last but not least the stored command parameters will be released.

6.2.2.1 Command Termination Philosophy

Due to the fact of the split design between ACI and SIM, the command termination of the OPEN CHANNEL and SEND DATA command is handled differently. As already mentioned earlier, ACI is responsible for the OPEN CHANNEL command and SIM for the SEND DATA command. Therefore both entities will terminate their own commands independently. A special case is the first SEND DATA command, which indicates an immediate channel establishment, after an on demand OPEN CHANNEL command. In this case SIM has to wait for the result of the establishment attempt, before it could terminate the command. Due to the fact that the SIM channel will always be connected at the end of the channel establishment procedure, ACI uses its request to pass the result to the SIM entity. In case of failure during the channel set up, ACI will close the open SIM channel and will use this request to pass the cause of the problem.

For the OPEN CHANNEL command termination by ACI, the elements *channel status*, *bearer description* and *buffer size* will only be included in the terminal response, if the channel establishment was successful. If a failure has occurred, only the *channel status* element will be part of the terminal response.

6.2.3 Open Channel State Machine

To fulfill the procedures of the channel establishment and support the asynchronous nature of command execution for the OPEN CHANNEL command, a state machine has been developed. See the following figure about all the states and conditions for transitions.

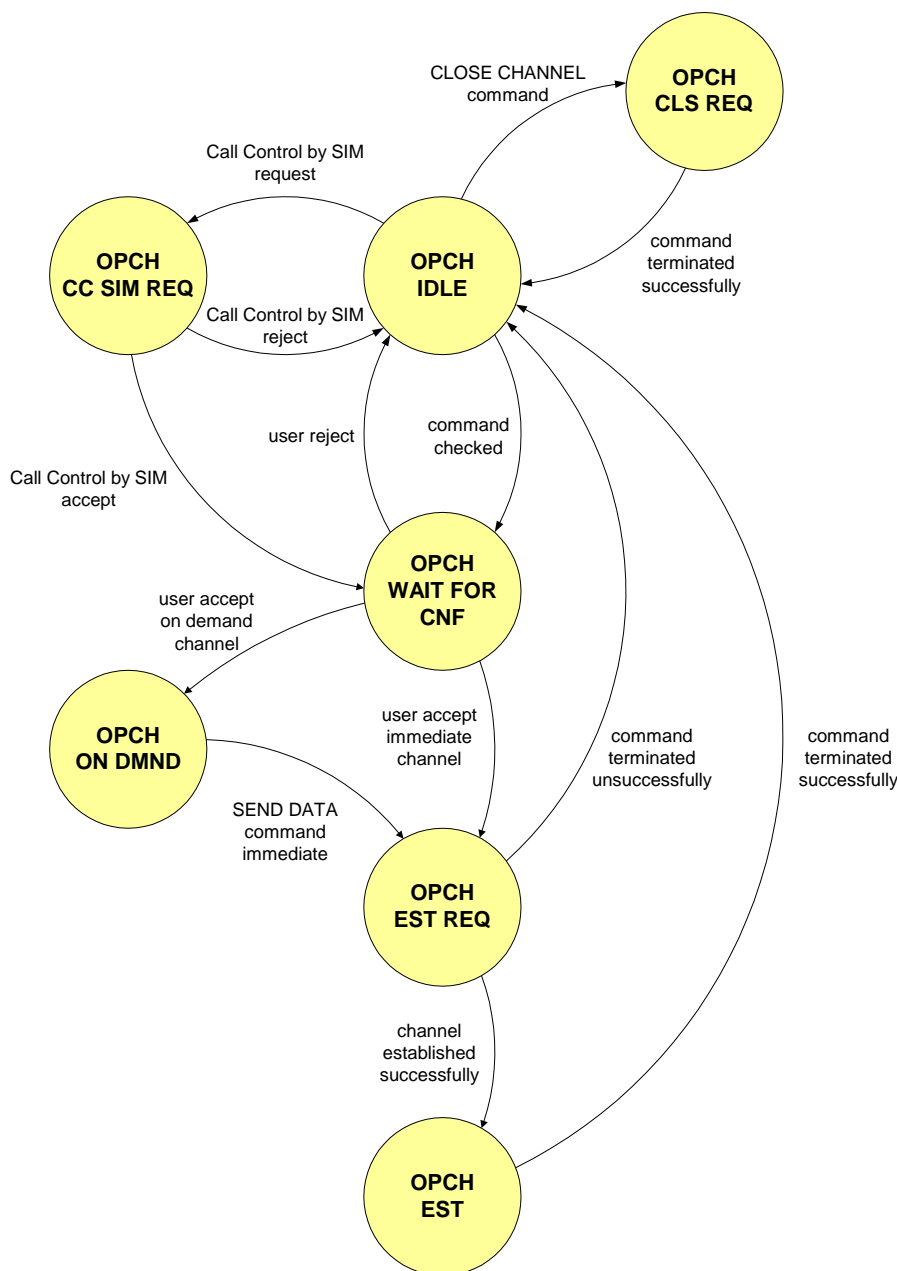


Figure 6: Open Channel State Machine

6.2.4 Channel Table

To keep track of the channel status and the corresponding channel parameters, a channel table has been designed. In fact the term channel table is a little bit exaggerating, because so far only one data channel for SAT is supported. But this concept has been chosen to be open for further extensions. A channel table entry consists of the following elements:

chnUsdFlg	// flags an used channel entry
chnRefId	// holds the reference to the channel bearer (call id for CSD, context id for GPRS)
chnType	// type of channel (CSD or GPRS)

chnTPL	// transport level (UDP or none)
linkStat	// current status of the link

Once a SAT channel has to be established, a channel table entry will be occupied. To be able to find and access the channel for further operation (e.g. close the channel) a reference identifier is stored. This is entry number in the call table for CSD channels and the PDP context id for GPRS channels. Therefore, to distinguish both references and for other purposes, the type of channel has to be stored too. The transport level type is important to activate/deactivate the UDP/IP data entities if requested by SAT. The current status of the data link between SAT (SIM) and the data channel is updated in the *linkStat* field of the channel table. See the following state diagram about the link states.

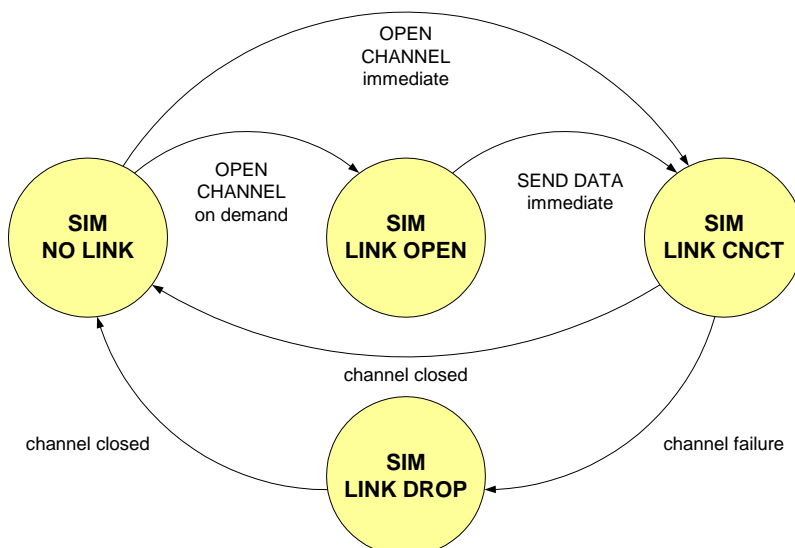


Figure 7: Link Status States

After the channel has been released, the channel table entry will be freed to be available for the next open channel command.

6.2.5 Channel Release

After a SAT data channel has been established there is only one other important thing for ACI within a channel life cycle, to release the channel. Actually there are two conditions, which finally lead to a channel release, the normal case and the abnormal case.

Normal Release:

The normal channel release will be initiated by SAT, by sending the CLOSE CHANNEL command to ACI. ACI will check whether the channel identifier matches the identifier of the open channel, this is actually the only important parameter that the command transports, from ACI point of view. After that ACI starts to release the data channel in the reverse order of channel establishment. The SIM channel will be closed first, followed by the data protocol entities and finally the network connection will be released. After the channel is released with the SIM entity, the command will be terminated successfully.

Another sort of normal release occurs, if the user clears down the data channel. If this happens during the establishment phase (OPEN CHANNEL or SEND DATA command), SAT will be informed about that conditions with an appropriate result cause in the terminal response for the command. In case of an immediate channel establishment, the user has to abort the AT A command in the usual way. If the establishment takes place after an on demand channel request, the user has to clear down the channel with the AT H or +CHUP command explicitly.

Abnormal Release:

An abnormal release occurs if one of the data protocol entities, the SIM entity or the network indicates a fail condition. In that case the channel will be released in the same way as described for the normal release. It depends always on what component of the data chain the problem has occurred. Finally all components will be deactivated and the SIM channel will be closed. Due to the fact that an abnormal release often occurs without being in the context of a SAT class e command, ACI indicates the channel release to SIM, when closing the channel. It is the task of the SIM entity to inform SAT about the broken channel when answering to the next SEND DATA or RECEIVE DATA command.

6.2.6 Event Handling

Two events are defined for SAT class e, the CHANNEL STATUS EVENT and the DATA AVAILABLE EVENT. Both events will be activated by SAT using the SET UP EVENT LIST command. The handling of this command is already implemented and basically not changed by this design. But what is true for the new SAT class e commands, is also true for the new events. They are splitted between ACI and SIM.

CHANNEL STATUS EVENT

The CHANNEL STATUS EVENT is handled by ACI. If SAT has activated this event, every unexpected channel release will be reported to SAT by an EVENT DOWNLOAD command with the current channel status. Due to the fact that a channel establishment is never unexpected, the event will not be sent if a channel has been opened.

DATA AVAILABLE EVENT

The DATA AVAILABLE EVENT is handled by SIM, as all data transmission issues are in responsibility of the SIM entity. Due to the fact that the SET UP EVENT LIST mechanism is handled in ACI, SIM will be informed whether this event is enabled or disabled. ACI will update the SIM entity after every SET UP EVENT LIST command that has been received.

7 Enhancements of SAT Message Catalogue

ACI uses CCD for coding and decoding of SAT responses and commands. Therefore all messages, which have to be exchanged between ACI and SAT, must be defined in the SAT message catalogue document [2]. This is the basis for CCD structures and tables. This section lists all changes that have to be made to extend the existing SAT message catalogue for SAT class c and e features.

7.1 New Messages

Open Channel

The OPEN CHANNEL command transports the request to open a new SAT data channel together with all parameters that are needed to establish the connection. The SAT specification [3GPP 11.14] shows two different layouts for a CSD and a GPRS channel. For the message description within the SAT message catalogue all elements of the two layouts have to be combined into one message. The reason is that there is only one SIM command tag for both variants of the OPEN CHANNEL command. Depending on the existing elements in the decoded command, it is the task of ACI to determine if a CSD or GPRS channel has to be opened. The command may include additional alpha or icon identifier, which are intended for display purposes.

Receive Data

With the RECEIVE DATA command SAT requests a certain amount of data from the temporary data buffer of the specified data channel. This command will not be decoded by ACI and is more important for the SIM entity, which handles the data transfer. It is listed here for completeness. The command may include additional alpha or icon identifier, which are intended for display purposes.

Send Data

The SEND DATA command transports data from SAT that has to be sent over the specified channel. This command will not be decoded by ACI and is more important for the SIM entity, which handles the data transfer. It is listed here for completeness. The command may include additional alpha or icon identifier, which are intended for display purposes.

Close Channel

To close an existing channel, SAT sends the CLOSE CHANNEL command to ACI. The command indicates the channel number that has to be released. The command may include additional alpha or icon identifier, which are intended for display purposes.

Get Channel Status

The GET CHANNEL STATUS command is used to request the complete status of all available data channels supported by ACI. The command is the trigger itself; no specific parameters are transported with this command.

Launch Browser

The LAUNCH BROWSER command transports the request to start the browser application together with all parameters that are needed to establish the connection. The command may include additional alpha or icon identifier, which are intended for display purposes.

7.2 New Structured Elements

Bearer Description

The bearer description consists of the bearer type identifier and either the set of bearer parameters for CSD or for GPRS. If the bearer type indicates default bearer, no bearer parameters are present.

CSD Bearer Parameter

Holds the bearer parameter speed, name and connection element for a CSD bearer.

GPRS Bearer Parameter

Holds the bearer parameter precedence, delay, reliability, peak throughput, mean throughput and PDP type for a GPRS bearer.

Channel Data

Holds a buffer for the channel data to be sent or the data that has been received over the channel.

Other Address / Destination Address

This element consists of the address type identifier and either a buffer for an IPv4 address or an IPv6 address.

Interface Transport Level

The ITL consists of the transport protocol type and a the port number to be used.

Channel Status Parameter

The channel status parameter element is part of the terminal response and consists of the link status, channel identifier and an additional status information.

Network Access Name

Holds the APN for the PDP context for a GPRS data channel.

Browser Identity

Determines what kind of browser has to be started.

URL

Holds the URL that has to be loaded by the browser application.

Provisioning File Reference

Holds a list of SIM file identifiers which should be used by the browser application.

Browser Termination Cause

The browser termination cause is part of the event download command and determines the source for termination of the browser application.

7.3 New Basic Elements

Due to the fact that these elements are simple, they are listed here only.

speed, name, connection element,

precedence, delay, reliability, peak throughput, mean throughput, PDP type,

network access name, bearer type,

channel data string, channel data length, channel identifier, channel status information, channel link status,

buffer size,

other address type, IPv4 address, IPv6 address,

transport protocol type, port number,

browser type, browser termination source,

URL value, provisioning file path identifier

8 Changes to existing Interfaces

8.1 ACI Functional Interface

The only function interface which will be affected by the SAT class C+E feature is that for `rAT_PercentSATA()`. In the past this function was solely used to indicate a pending setup call request by SAT. For the SAT class C+E feature the parameters for this function will be enhanced, to distinguish between call setup, circuit switched data setup and GPRS context activation. The additional information is packet into a separate structure of type `T_ACI_SATA_ADD`.

`T_ACI_SATA_ADD`

Type Declaration:

struct type	element	element type	comment
<code>T_ACI_SATA_ADD</code>	<code>chnType</code>	<code>T_ACI_SATA_CHN</code>	type of channel information
	<code>chnEst</code>	<code>T_ACI_SATA_EST</code>	type of establishment information

`T_ACI_SATA_CHN`

Type Declaration:

enum type	value	symbolic constant	comment
<code>T_ACI_SATA_CHN</code>	0	<code>SATA_CT_NO</code>	no channel type given
	1	<code>SATA_CT_VOICE</code>	channel type voice
	2	<code>SATA_CT_CSD</code>	channel type CSD
	3	<code>SATA_CT_GPRS</code>	channel type GPRS

`T_ACI_SATA_EST`

Type Declaration:

enum type	value	symbolic constant	comment
<code>T_ACI_SATA_EST</code>	0	<code>SATA_EST_NO</code>	no establishment type given
	1	<code>SATA_EST_IM</code>	immediate channel establishment
	2	<code>SATA_EST_OD</code>	on-demand channel establishment

The new functional interface for the `rAT_PercentSATA()` function will look like the following:

`rAT_PercentSATA ()` – Alert about Sim Application Toolkit Pending Call Setup

Command Reference:

none

Function Definition:

```
void rAT_PercentSATA (SHORT cld, LONG rdTimeout_ms, T_ACI_SATA_ADD *addParm);
```

Parameters:

name	comment	
<code>cld</code>	call id	IN
<code>rdTimeout_ms</code>	redial timeout in ms	IN
<code>addParm</code>	additional parameters to distinguish channel type and mode of establishment	IN

Return:

none

Function Group:

none

Description:

The function `rAT_PercentSATA` informs the application about a channel setup, which is initiated by the SIM application toolkit. The value `cld` represents the channel identifier. In case of a circuit switched call (`addParm.chnType` is set to `SATA_CT_VOICE` or `SATA_CT_CSD`) the `cld` indicates the call identifier within the call table. In case of a GPRS channel (`addParm.chnType` is set to `SATA_CT_GPRS`) the `cld` indicates the context identifier of the defined PDP context. The value `rdlTimeout_ms` is the radial timeout in units of ms. The contents of `addParm` gives information to distinguish between the different channel types. Within `addParm` the setting of `chnType` shows the type of channel and `chnEst` indicates whether the channel should be established immediately or on demand. The call is accepted by using the AT A command (`sAT_A` function) and is rejected by using the AT H command (`sAT_H` function).

8.2 AT Interface

The SAT class E feature affects only one AT command and one unsolicited result code. To indicate a Open Channel request to the user the existing result code `%SATA:` will be used. In the past this result code was solely used to indicate call set up requests from SIM application toolkit. To separate call set up requests from open channel request additional parameters are necessary. To be compliant with the existing version of the AT interface, these additional parameters for `%SATA:` have to be enabled explicitly with the `%SATC` command. This command was already in use to enable SAT unsolicited results in general. The `<n>` parameter of the `%SATC` command will be extended to enable the additional `%SATA` parameters for SAT class E. This will only effect the `%SATA` result code. The other SAT result codes will be also enabled but remain the same. The additional parameters show the type of channel that is requested and the mode of establishment. The new command description for `%SATC` will look like the following:

x.y Configuration for SIM application toolkit %SATC

Command	Possible response(s)
<code>%SATC=<n>,<satPrfl></code>	
<code>%SATC?</code>	<code>%SATC: <n>,<satPrfl></code>
<code>%SATC=?</code>	<code>%SATC: (list of supported <n>s),(<prflLen>)</code>

Description

This command refers to the SIM application toolkit download mechanism, which is used to indicate to the SIM the features that the ME is capable of. The different features that are possible for a proactive SIM card are summarized by a table called a profile, refer to GSM 11.14 for more details. Conda's ACI, SMS and SIM modules already implement some of these features. Therefore the profile that is indicated by `<satPrfl>` will be combined with the existing one. The current profile setting could be displayed using the read command. `<n>` is used to enable/disable the presentation of unsolicited notification result codes from TA to TE.

If `<n>=1` and one of the following conditions have occurred, the respective unsolicited result is sent to TE.

- A command received from the SIM that is not handled by ME is indicated to TE by `%SATI: <satCmd>`.
- The result to an envelope command, which was sent by TE, is indicated using the result `%SATE: <satRsp>`. For more information regarding the sending of envelope commands to SIM, please refer to the `%SATE` command description.
- If SIM application toolkit tries to set up a call using the Set Up Call feature as described in GSM 11.14, and the conditions for the call were checked by ME successfully, the request is indicated to TE using the result `%SATA: [<rdl>]`. Using the accept command A, ME tries to establish the call, otherwise the hook-on command H rejects the pending SAT call and sends the respective response to SIM.

- In general, commands or responses sent by ME to SIM or commands handled by ME are indicated to TE using the result %SATN: <satNtfy>. With these notifications, TE shall be able to indicate appropriate messages to a user.

Setting <n>=2 enables the unsolicited result code %SATA: to output additional parameters which show details about the type of channel and the mode of channel establishment. This setting is recommended if SAT class E is available. All other SAT result codes are enabled too and will behave as described above.

- If SIM application toolkit tries to set up a call or to establish a data channel using the Set Up Call or Open Channel feature as described in GSM 11.14 and the conditions were checked by ME successfully, the request is indicated to TE using the result %SATA: [<rd1>], [<type>], [<est>]. To accept or reject the channel establishment to user can proceed as mentioned above.

Defined Values

<n>: (parameter sets/shows the result code presentation status in the TA)

<satPrfl>: string type; SIM application toolkit profile (hexadecimal format; refer +CSCS) starting with first byte of the profile.

<satCmd>: string type; SIM application toolkit command (hexadecimal format; refer +CSCS) starting with command tag.

<satRsp>: string type; SIM application toolkit response (hexadecimal format; refer +CSCS) starting with first byte of response data.

<satNtfy>: string type; commands or responses sent by ME to SIM or commands handled by ME (hexadecimal format; refer +CSCS) starting with first byte of response data or command tag.

<rd1>: integer type; if a pending SIM application toolkit command is alerted to TE using result %SATA:, the value of <rd1> indicates the redial timeout for the call in unit of milliseconds.

<type>: integer type; determines the type of channel that is requested by SIM application toolkit command.

- 1 voice channel
- 2 circuit switched data channel, CSD
- 3 packet switched data channel, GPRS

<est>: integer type; determines the mode of channel establishment

- 1 immediate channel establishment
- 2 on demand channel establishment

9 Integration with Existing Software

9.1 Extension of Existing Data Types for SAT Class E

Some already defined data types have to be extended to support the SAT class e features. The extension has to be made to be compliant with the existing implementation. The new elements within the already defined types are marked in **bold font**.

9.1.1 Module psa_cc.h

9.1.1.1 T_CC_CLST

```
typedef enum                                /* call status */
{
    NO_VLD_CS = 0,                        /* not a valid call status */
    CS_IDL,                               /* call idle */
    CS_ACT_REQ,                           /* call active request */
    CS_ACT,                               /* call active */
}
```

```

    CS_HLD_REQ,                /* call hold request */
    CS_HLD,                    /* call held */
    CS_DSC_REQ,                /* call disconnect request */
    CS_CPL_REQ,                /* call completion request */
    CS_MDF_REQ,                /* call modification request */
    CS_SAT_REQ,                /* call SAT request */
    CS_SAT_CSD_REQ            /* call SAT open CSD channel request */
} T_CC_CLST;

```

Description:

The new call status for a CSD request has to be introduced.

9.1.2 Module psa_sat.h**9.1.2.1 T_SAT_SHRD_PRM**

```

typedef struct SATShrdParm
{
    UBYTE          owner;                /* identifies the used set */
    UBYTE          ntfty;                /* user notification */
    T_SAT_SET_PRM  setPrm[OWN_MAX];     /* possible sets */
    T_SAT_CC_PRM   SIMCCParm;           /* call control by SIM parameter */
    T_SAT_CAP_PRM  capParm;             /* cap request parameter */
    T_SAT_CMD_DET  cmdDet;              /* command details */
    LONG           dur;                 /* redial duration */
    UBYTE          *stkCmd;              /* points to STK command */
    USHORT         stkCmdLen;           /* length of STK command */
    USHORT         fu_rsc;              /* result code of FILE UPDATE */
    BUF_cmd_prms   stkCmdPrm;          /* proactive SIM command parameter */
    T_SIM_TOOLKIT_CNF *stkCnfPrim;      /* pending result primitive */
    T_SIM_FILE_UPDATE_IND *fu_ind;      /* actual FILE UPDATE primitive */
    T_SAT_FU_FUNC  fu_func[MAX_FU_OPS]; /* registered FILE UPDATE handlers */
    T_SAT_FU_FUNC  fu_func_notif;       /* registered FILE UPDATE NOTIFY handler */
    T_SAT_EVENT     event;              /* flags events to be monitored for SAT */
    SHORT          SentUSSDid;          /* ssId of USSD sent */
    BOOL           USSDterm;            /* flags if user is aiming at terminating ussd */
    /*
    UBYTE          run_at_id;            /* srce id of source where run at cmd is
    processed */
    UBYTE          opchStat;             /* open channel status */
    UBYTE          opchType;            /* open channel type */
    void           *opchPrm;            /* points to open channel parameters */
    UBYTE          opchPrmMdf;          /* flags bearer parameter modification */
    UBYTE          opchAcptSrc;         /* open channel source identifier */
    UBYTE          opchCCMdfy;          /* flags open channel call control modification */
    /*
    UBYTE          gprsNtwCs;            /* gprs network cause */
    T_SAT_CHN_NTRY chnTb;              /* channel table, must be an array if more than
    one
                                         channel will be supported */
} T_SAT_SHRD_PRM;

```

Description:

The shared parameter structure of PSA SAT has to be extended by some variables that keep track of the current channel status and command execution. The channel table is also part of this extension. Nevertheless it is limited to one entry for now.

9.1.3 Module psa_sim.h**9.1.3.1 T_ACI_SAT_TERM_RESP**

```

typedef struct
{
    UBYTE          addLen;
    UBYTE          *add;
    UBYTE          add_content; /* when addLen = 1 */
    UBYTE          *resCC;
    T_text         *text;
    UBYTE          *at_resp;
    USHORT         at_resp_count;
    UBYTE          chnStat;

```

```

    UBYTE    bearDesc;
    UBYTE    bufSize;
} T_ACI_SAT_TERM_RESP;

```

Description:

The list of flags to build the terminal response will be extended to switch on channel status, bearer description and/or buffer size elements.

9.1.3.2 T_SIM_SHRD_PRM

```

typedef struct SIMShrdParm
{
    UBYTE    owner;                /* identifies the used set */
    SHORT    aId;                 /* access identifier */
    T_SIM_SET_PRM setPrm[OWN_MAX]; /* possible sets */
    T_SIM_ACC_PRM atb[ACC_MAX];    /* table of access parameter */

    UBYTE    PINStat;            /* status of PIN requirement */
    UBYTE    pn1Cnt;             /* PIN 1 counter */
    UBYTE    pn2Cnt;             /* PIN 2 counter */
    UBYTE    pk1Cnt;             /* PUK 1 counter */
    UBYTE    pk2Cnt;             /* PUK 2 counter */
    UBYTE    pn1Stat;            /* PIN 1 status */
    UBYTE    pn2Stat;            /* PIN 2 status */
    UBYTE    crdPhs;             /* phase of card */
    UBYTE    SIMStat;            /* status of SIM card */
    UBYTE    PEDStat;            /* PIN1 enable/disable status */
    UBYTE    crdFun;             /* SIM card functionality */
    UBYTE    srvTab[SRV_TAB_LEN]; /* SIM service table */
    T_imsi_field imsi;           /* IMSI */
    USHORT    rslt;              /* result of SIM operation */
    UBYTE    synCs;              /* SIM synchronisation cause */
    UBYTE    ciStat;             /* current ciphering indicator state */
    UBYTE    savedCiStat;        /* saved ciphering indicator state */
    int    fuRef;                /* Reference for File Update */
    T_SIM_DTI_CH_PRM *sim_dti_chPrm; /* points to sim dti channel parameters */
} T_SIM_SHRD_PRM;

```

Description:

The new structure of dti channel parameters has to be introduced.

9.2 Modification of Existing Functions for SAT Class E

Some existing ACI functions have to be extended or modified to handle the SAT Class e functionality. The following is a list of all modules and functions, which have to be modified. A short description will indicate the impact of the modification.

9.2.1 Module ati_sat.c

9.2.1.1 rCI_PercentSATA

- adjust interface changes.
- output of extended SAT class e result parameters.

9.2.1.2 setatPercentSATC

- support enabling of extended SAT class e result parameters.

9.2.2 Module cmh_ccr.c

9.2.2.1 cmhCC_callConnected

- in case of a failure cmhSAT_OpChnCSDDown() will be called to indicate this to the SAT command handler.
- the function cmhSAT_OpChnChckCSD() has to be used to check for SAT open channel context.

9.2.2.2 cmhCC_callDisconnected

- in case of a failure cmhSAT_OpChnCSDDown() will be called to indicate this to the SAT command handler.
- in case of a on demand channel establishment attempt, the final AT response has to be suppressed because the call setup was not within a AT command context

9.2.2.3 cmhCC_L2R_or_TRA_Activated

- the function cmhSAT_OpChnChkCSD() has to be used to check for SAT open channel context
- the function cmhSAT_OpChnSIMCnctReq() has to be called to request for a SIM channel connection.

9.2.2.4 cmhCC_PPP_Established

- the function cmhSAT_OpChnChkCSD() has to be used to check for SAT open channel context
- install own callback function for UDP configuration.

9.2.2.5 cmhCC_PPP_Terminated

- the function cmhSAT_OpChnChkCSD() has to be used to check for SAT open channel context
- install own callback function for UDP deactivation.

9.2.2.6 cmhCC_RA_Deactivated

- in case of a failure cmhSAT_OpChnCSDDown() will be called to indicate this to the SAT command handler.

9.2.2.7 cmhCC_RA_Failed

- cmhSAT_OpChnCSDDown() will be called to indicate the fail condition to the SAT command handler.
- the function cmhSAT_OpChnChkCSD() has to be used to check for SAT open channel context

9.2.3 Module cmh_ccs.c

9.2.3.1 sAT_A

- additional check for pending CSD request
- additional check for a pending GPRS context, in that case call the function cmhSAT_UserAcptCntxt() to indicate this to the SAT command handler.

9.2.3.2 sAT_H, sAT_PlusCHUP

- additional check for pending CSD request
- additional check for a pending GPRS context, in that case call the function cmhSAT_UserRejCntxt() to indicate this to the SAT command handler.

9.2.4 Module cmh_gmmr.c

9.2.4.1 cmhGMM_Suspend / cmhGMM_Resume

- call cmhSAT_OpChnGPRSStat() to indicate the suspend or resume state of GPRS.

9.2.5 Module cmh_l2rs.c

0.1.1.1 cmhL2R_Deactivate

- start deactivation of entity L2R if internal dti connection.

0.1.1.2 cmhTRA_Deactivate

- disconnection of dti connection with entity TRA and deactivation of entity TRA.

9.2.6 Module cmh_satf.c

9.2.6.1 cmhSAT_ChckRedial

- distinguish between SET UP CALL and OPEN CHANNEL command.
- adjust the function parameters to be more generic and usable for both kind of commands.

9.2.6.2 cmhSAT_CheckSetEventResp

- extend to 32 biteventlist

9.2.6.3 cmhSAT_SetupCall

- extend RAT_SATA callback by additional parameters.

9.2.6.4 cmhSAT_setupEvent_Perform

- in case of DATA AVAILABLE event, call passim_EvDataAvail()

9.2.6.5 cmhSAT_setupEvent_Test

- enhancement of event mechanism for CHANNEL STATUS and DATA AVAILABLE event
- extend eventlist from 8 bit to 32 bit

9.2.6.6 cmhSAT_NtwErr

- return code changed from void to UBYTE. The return code indicates, whether the caller should send an AT response if applicable or if the response should be suppressed. This is needed in case of an on demand channel establishment failure when the call setup is not within a AT command context
- in case of a SEND DATA command, the SIM channel has to be closed.
- in case of OPEN CHANNEL command set the channel status flag for the terminal response, free the channel table entry and call cmhSAT_cleanupOpchPrms().

9.2.6.7 cmhSAT_ResCapDecode

- extend RAT_SATA callback by additional parameters.

9.2.6.8 cmhSAT_SetupCallAfterCCRes

- extend RAT_SATA callback by additional parameters.

9.2.6.9 cmhSAT_UserAcptCall

- the function has to distinguish between SET UP CALL and OPEN CHANNEL command.
- call owner will be reseted to OWN_SAT.
- in case of a on demand channel establishment, a SIM data channel has to be opened.
- in case of a immediate channel establishment, check for temporary problems, which will hinder the command execution.
- set WAP flag and PPP parameters in case of UDP as the interface transport layer.

9.2.6.10 cmhSAT_UserClear

- in case of a SEND DATA command, the SIM channel has to be closed.
- in case of OPEN CHANNEL command set the channel status flag for the terminal response, free the channel table entry and call cmhSAT_cleanupOpchPrms().

9.2.6.11 cmhSAT_UserRejCall

- in case of CSD establishment request, set channel status flag for terminal response.
- at the end call cmhSAT_cleanupOpChnPrms()

9.2.7 Module cmh_satr.c

9.2.7.1 cmhSAT_ResCalCntrlBySIM()

- in case the call set up will be not allowed by SIM, check for a CSD request and indicate a fail condition by calling cmhSAT_OpChnFailed().
- in case the call has been modified into a SS transaction, check for a CSD request and indicate a fail condition by calling cmhSAT_OpChnFailed().
- in case the call has not been modified, check for a CSD request and call cmhSAT_OpChnAlert().
- extend RAT_SATA callback by additional parameters.

9.2.7.2 cmhSAT_ResSSCntrlBySIM()

- extend RAT_SATA callback by additional parameters.

9.2.7.3 cmhSAT_ResUSSDCntrlBySIM()

- extend RAT_SATA callback by additional parameters.

9.2.8 Module cmh_sats.c

9.2.8.1 sAT_PercentSATT

- additional check for pending OPEN CHANNEL or SEND DATA commands.
- free channel table entry and call cmhSAT_cleanupOpChnPrms()

9.2.9 Module cmh_smf.c

9.2.9.1 cmhSM_IP_activate_cb

- check activation status SAT open channel command context and switch to own callback functions. cmhSAT_OpChnUDPDeactGprs() , cmhSAT_OpChnUDPCnfGprs().

9.2.10 Module cmh_smr.c

9.2.10.1 cmhSM_Activated

- the function gaci_RAT_caller replaces the call to RAT_OK.

9.2.11 Module gaci.c

9.2.11.1 gaci_RAT_caller

- will be extended to indicate the current GPRS status to ACI

9.2.12 Module gaci_srcc.c

srcc_delete_dti_id

- delete existing source id for link between SIM entity and SNDCP entity

srcc_new_dti_id

- create new source id for link between SIM entity and SNDCP entity

srcc_reserve_sources

- check if new link between SIM entity and SNDCP entity is possible

9.2.13 Module dti_manager.c

0.1.1.3 connect_entity

- added connect of following entities: SIM and UDP (under WAP)

0.1.1.4 disconnect_entity

- disconnect of dti connection for L2R, UDP and SIM entity updated and/or added

0.1.1.5 dti_mng_remove

- remove of dti connection for L2R entity updated

9.2.14 Module psa_l2rs.c

psaL2R_Deactivate

- add new case L2R_DEACTIVATED for release of DTI connection between SIM entity and L2R entity

psaL2R_Enable

- send L2R_DTI_REQ for link between SIM entity and L2R entity

9.2.15 Module psa_l2rp.c

psa_l2r_disc_ind

- L2R_DISC_IND causes disconnecting of DTI connection between SIM entity and L2R entity

psa_l2r_dti_cnf

- after L2R_DTI_CNF dti manager starts to build link with SIM entity in case SAT Class e

psa_l2r_error_ind

- disconnect DTI connection between SIM entity and L2R entity is later handled by dti manager

9.2.16 Module psa_satf.c

9.2.16.1 psaSAT_BuildEnvEventDwn

- extend eventdownload command to transport the CHANNEL STATUS event

9.2.16.2 psaSAT_dasmMECmd

- extend command handling to support the OPEN CHANNEL command.

9.2.16.3 psaSAT_InitTrmResp

- initialize new flags for channel status, bearer description and buffer size.

9.2.16.4 psaSAT_SendTrmResp

- fill in data for channel status, bearer description and buffer size.

9.2.17 Module psa_satp.c

9.2.17.1 psa_sim_toolkit_ind

- extend the command distribution for OPEN CHANNEL, CLOSE CHANNEL, SEND DATA, RECEIVE DATA and GET CHANNEL STATUS commands.

9.2.18 Module `psa_smf.c`

0.1.1.6 `psaSNDTCP_Dti_Req`

- connect an activated PDP context with the SIM entity.

9.2.19 Module `psa_tras.c`

0.1.1.7 `psaTRA_Dti_Req`

- send TRA_DTI_REQ for connection between SIM entity and TRA entity in case SAT Class e

9.2.20 Module `psa_udpap.c`

0.1.1.8 `psa_udpa_config_cnf`

- after received UDPA_CONFIG_CNF dti manager starts to build or to disconnect link between UDP entity and SIM entity in case SAT Class e

9.2.21 Module `psa_udpas.c`

0.1.1.9 `psaUDPA_Config`

- add new parameters entity_name, tui, layer, c_id and dti_conn in primitive UDPA_CONFIG_REQ
- inform entity UDP about connection with low layer IP

9.2.22 Module `smi_dmy.c`

9.2.22.1 `rAT_PercentSATA`

- adjust interface changes.

10 New Functions and Data Structures

10.1 Predefined Values

10.1.1 Module `cmh_sat.h`

```
#define SAT_GPRS_INV_CAUSE 0
```

10.1.2 Module `psa_sat.h`

```
#define MAX_SAT_CHANNEL      1      /* maximum number of channels available */  
#define TPL_NONE             0      /* no transport layer */  
#define TPL_DONT_CARE        0xFF
```

10.1.3 Module `psa_tcpip.h`

```
#define TCPIP_CONNECTION_TYPE_GPRS_SAT 0x04  
#define TCPIP_CONNECTION_TYPE_CSD_SAT  0x08
```

10.2 New Data Types Introduced for SAT Class E

10.2.1 Module `aci_cmh.h`

10.2.1.1 `T_ACI_SATA_ADD`

```
typedef struct  
{  
    T_ACI_SATA_CHN chnType;
```

```
T_ACI_SATA_EST chnEst;
}
T_ACI_SATA_ADD;
```

Description:

A new type to transport channel type and mode of channel establishment as additional parameters for %SATA unsolicited result code. Module cmh_sat.h

10.2.1.2 T_ACI_SATA_CHN

```
typedef enum
{
    SATA_CT_NO = 0,          /* no channel type given */
    SATA_CT_VOICE,          /* channel type voice */
    SATA_CT_CSD,            /* channel type CSD */
    SATA_CT_GPRS            /* channel type GPRS */
} T_ACI_SATA_CHN;
```

Description:

Definition of values to indicate the type of call for %SATA unsolicited result code.

10.2.1.3 T_ACI_SATA_EST

```
typedef enum
{
    SATA_EST_NO = 0,        /* no establishment type given */
    SATA_EST_IM,           /* immediate channel establishment */
    SATA_EST_OD            /* on-demand channel establishment */
} T_ACI_SATA_EST;
```

Description:

Definition of values to indicate the mode of channel establishment for %SATA unsolicited result code.

10.2.1.4 T_ACI_SATC_STAT

```
typedef enum
{
    SATC_DIS = 0,          /* unsolicited SAT output disabled */
    SATC_ENA,             /* standard unsolicited SAT output enabled */
    SATC_ENA_CL_E        /* class E unsolicited SAT output enabled */
} T_ACI_SATC_STAT;
```

Description:

Definition of values to set the behavior for unsolicited SAT result codes.

10.2.1.5 T_SAT_GPRS_CB_STAT

```
typedef enum
{
    SAT_GPRS_ACT = 0,
    SAT_GPRS_DEACT,
    SAT_GPRS_ATT_FAILED,
    SAT_GPRS_ACT_FAILED,
    SAT_GPRS_SUSPEND,
    SAT_GPRS_RESUME
} T_SAT_GPRS_CB_STAT;
```

Description:

Values defined to indicate the GPRS status, when establishing a GPRS channel

10.2.2 Module psa_sat.h

10.2.2.1 T_SAT_CHN_NTRY

```
typedef struct
{
    UBYTE chnUsdFlg;          /* flags a used channel */
    SHORT chnRefId;          /* refers to call id for CSD, or CID for GPRS */
    UBYTE chnType;           /* channel type (CSD/GPRS) */
}
```

```

        UBYTE chnTPL;                /* channel transport layer */
        UBYTE lnkStat;               /* sim link status */
    } T_SAT_CHN_NTRY;

```

Description:

Group of elements, which describe the characteristic data of an SAT data channel. Will be used as an entry of the channel table.

10.2.2.2 T_SAT_CSD_PRM

```

typedef struct
{
    UBYTE def_bear_prm; /* true or false for default parameters */
    T_csd_bear_prm csd_bear_prm; /* if no def. parms. use CSD parms.*/
    UBYTE v_dur; /* duration 1 avail.*/
    T_dur dur; /* duration 1 */
    UBYTE v_dur2; /* duration 2 avail.*/
    T_dur2 dur2; /* duration 2 */
    UBYTE v_other_addr; /* other addr. avail.*/
    T_other_addr other_addr; /* other addr.*/
    UBYTE v_log; /* login name avail.*/
    T_text log; /* login name */
    UBYTE v_pwd; /* password avail.*/
    T_text2 pwd; /* password */
    UBYTE v_itl; /* if transp level avail.*/
    T_if_transp_lev itl; /* if transp level */
    UBYTE v_dda; /* data dest. addr. avail.*/
    T_data_dest_addr dda; /* data dest. addr.*/
} T_SAT_CSD_PRM;

```

Description:

This type is used to store necessary CSD parameters of the SAT command, which are needed at a later stage of channel establishment.

10.2.2.3 T_SAT_GPRS_PRM

```

typedef struct
{
    UBYTE def_bear_prm; /* true or false for default parameters */
    T_gprs_bear_prm gprs_bear_prm; /* if no def. parms. use GPRS parms.*/
    UBYTE v_apn; /* apn avail.*/
    UBYTE c_apn; /* apn length */
    UBYTE apn[MAX_SAT_APN_LEN]; /* apn */
    UBYTE v_other_addr; /* other addr. avail.*/
    T_other_addr other_addr; /* other addr.*/
    UBYTE v_itl; /* if transp level avail.*/
    T_if_transp_lev itl; /* if transp level */
    UBYTE v_dda; /* data dest. addr. avail.*/
    T_data_dest_addr dda; /* data dest. addr.*/
} T_SAT_GPRS_PRM;

```

Description:

This type is used to store necessary GPRS parameters of the SAT command, which are needed at a later stage of channel establishment.

10.2.2.4 T_SAT_OPCH_STAT

```

typedef enum
{
    OPCH_IDLE = 0, /* open channel status idle */
    OPCH_CC_SIM_REQ, /* open channel status CC by SIM request */
    OPCH_WAIT_CNF, /* open channel status wait for confirmation */
    OPCH_ON_DMND, /* open channel status on demand */
    OPCH_EST_REQ, /* open channel establishment request */
    OPCH_CLS_REQ, /* channel close request */
    OPCH_EST, /* channel established */
    OPCH_SUSP, /* channel suspended */
    OPCH_NONE
} T_SAT_OPCH_STAT;

```

Description:

Values for the open channel state machine.

10.2.2.5 T_SAT_SIM_LNK_STAT

```
typedef enum
{
    SIM_NO_LINK = 0,           /* no link established */
    SIM_LINK_OPEN,           /* link opened */
    SIM_LINK_CNCT,           /* link connected */
    SIM_LINK_DROP           /* link dropped */
} T_SAT_SIM_LNK_STAT;
```

Description:

Values to describe the link status.

10.2.3 Module psa_sim.h

10.2.3.1 T_SIM_DTI_CH_PRM

```
typedef struct
{
    T_SIM_SAT_CHN sat_chn_prm;
    void (*cb)(UBYTE dtiConn, UBYTE chnId);
    UBYTE dti_id;
} T_SIM_DTI_CH_PRM;
```

Description:

Definition of values for establishment of DTI connection between SIM entity and an other entity: MMI callback function and dti connection identifier.

10.2.3.2 T_SIM_SAT_CHN

```
typedef struct
{
    UBYTE dtiConn;           /* connection qualifier */
    UBYTE dtiUnit;           /* dti connection unit */
    UBYTE chnId;             /* channel id */
    UBYTE genRes;            /* general result */
    UBYTE addRes;            /* additional result */
} T_SIM_SAT_CHN;
```

Description:

Definition of values for channel establishment: status of connection, partner entity, desired channel identifier.

10.3 New Functions for SAT Class E

10.3.1 Module ati_ret.c

10.3.1.1 search_SATSrclId

Prototype:

```
UBYTE search_SATSrclId (void);
```

Parameters:

none

Return:

SAT source identifier

Description:

This function search in pool of ACI source identifiers the SAT source id.

10.3.2 Module cmh_satf.c

10.3.2.1 cmhSAT_cleanupOpChnPrms

Prototype:

```
void cmhSAT_cleanupOpChnPrms( void );
```

Parameters:

none

Return:

none

Description:

This function cleans the channel table entry and frees allocated memory.

10.3.2.2 cmhSAT_CloseChannel

Prototype:

```
BOOL cmhSAT_CloseChannel( void );
```

Parameters:

none

Return:

BOOL

Description:

This function will be called after reception of a CLOSE CHANNEL command. The user will be notified about the reception of the command and then the channel status is checked. If a SIM data channel is open or connected, the SIM channel will be released first. Otherwise the ACI will proceed with disconnecting the bearer. The function will return TRUE if the command will be executed.

10.3.2.3 cmhSAT_cnvrtpAPN2NetworkAdr

Prototype:

```
void cmhSAT_cnvrtpAPN2NetworkAdr( UBYTE *apn, UBYTE c_apn, UBYTE *dom_name );
```

Parameters:

apn	: points to the APN string
c_apn	: indicates the length of the APN string
dom_name	: points to destination for converted domain name string, null terminated

Return:

none

Description:

This function is used to convert an APN with a given length into a null terminated domain name (dot separated).

10.3.2.4 cmhSAT_GetChannelStatus

Prototype:

```
BOOL cmhSAT_GetChannelStatus( void );
```

Parameters:

none

Return:

BOOL

Description:

This function will be called after reception of a GET CHANNEL STATUS command. The user will be notified about the reception of the command. Then the function will send the current channel status and terminate the command. The function will always return FALSE because command execution ends within the context of the function.

10.3.2.5 cmhSAT_OpenChannelReq

Prototype:

```
BOOL cmhSAT_OpenChannelReq( T_OPEN_CHANNEL *opchCmd );
```

Parameters:

opchCmd : pointer to the decoded open channel command parameters

Return:

BOOL

Description:

This function will be called after reception of an OPEN CHANNEL command. First of all some basic conditions are checked, which will lead to the rejection of the command. If none of these conditions apply, the function will separate the handling for CSD and GPRS channel requests. In case of a CSD channel a call table entry will be allocated and filled with necessary parameters. For a GPRS channel an appropriate PDP context will be defined. A channel table entry will be occupied to store the current channel status. If Call Control by SIM is activated and allocated, all CSD channels will be checked by SIM. The command execution will be paused until the result is available. If not or for all GPRS channel request the function will finally alert the user about a pending SAT channel request. The function will return TRUE if the command will be executed.

10.3.2.6 cmhSAT_OpChnAlert

Prototype:

```
void cmhSAT_OpChnAlert( SHORT cld );
```

Parameters:

cld : channel or context identifier for the pending channel request

Return:

none

Description:

This function is used to alert the user about the pending open channel request.

10.3.2.7 cmhSAT_OpChnChckBear

Prototype:

```
void cmhSAT_OpChnChckBear(void);
```

Parameters:

none

Return:

none

Description:

This function is used to compare the originally set bearer parameters with the actual one. A flag will be set, which indicates a modification of bearer parameters during the channel establishment. This has to be considered when sending the terminal result cause to SAT.

10.3.2.8 cmhSAT_OpChnChckCSD

Prototype:

```
BOOL cmhSAT_OpChnChckCSD( UBYTE tpl );
```

Parameters:

tpl : transport protocol layer

Return:

BOOL

Description:

This function is used to check whether a CSD channel request for the given transport protocol layer is pending or not.

10.3.2.9 cmhSAT_OpChnChkTmpProblem

Prototype:

```
BOOL cmhSAT_OpChnChkTmpProblem( void );
```

Parameters:

none

Return:

BOOL

Description:

This function checks for a temporary situation, which prevents the OPEN CHANNEL / SEND DATA command to be completed. If such a condition exists, the command will be aborted with a suitable terminal response and the function returns TRUE. If necessary a cleanup of parameters will be performed.

10.3.2.10 cmhSAT_OpChnClose

Prototype:

```
void cmhSAT_OpChnClose( UBYTE dtiConn, UBYTE chnId );
```

Parameters:

dtiConn	: SIM channel status
chnId	: channel identifier

Return:

none

Description:

This callback function will be called, when the SIM entity has tried to close a SIM channel. If the closing was in context with an OPEN CHANNEL command, a channel failure will be indicated terminate the command. The command parameters will be cleaned up. If the close operation was in context with a CLOSE CHANNEL command the command will be terminated successfully. If the bearer is still in use, the bearer will be released immediately.

10.3.2.11 cmhSAT_OpChnCnct

Prototype:

```
void cmhSAT_OpChnCnct( UBYTE dtiConn, UBYTE chnId );
```

Parameters:

dtiConn	: SIM channel status
chnId	: channel identifier

Return:

none

Description:

This callback function will be called, when the SIM entity has tried to connect a SIM channel. If the connection was successful and in context with an OPEN CHANNEL command, the SAT command and the user command to accept the channel will be terminated with a positive result. The command parameters will be cleaned up. If the connection was not successful and the SIM channel is still open, the channel will be closed immediately. If not and the bearer is still in use, the bearer will be released immediately.

10.3.2.12 cmhSAT_OpChnCSDDown

Prototype:

```
void cmhSAT_OpChnCSDDown( SHORT cId, UBYTE tpl );
```

Parameters:

cId	: call identifier
tpl	: transport protocol layer

Return:

none

Description:

This function will be called if a condition occurs, when the CSD connection will break down. The function checks whether the breakdown is in context with an established SAT channel. If that is true, a channel failure will be indicated to proceed accordingly with the current command context.

10.3.2.13 cmhSAT_OpChnFailed

Prototype:

```
void cmhSAT_OpChnFailed( UBYTE cause, T_ACI_SAT_TERM_RESP *resp_data );
```

Parameters:

cause	: cause for SAT terminale response
resp_data	: pointer to additional terminal response parameters

Return:

none

Description:

This function will be called if a failure for a SAT data channel has occurred. If the failure occurred in context with the OPEN CHANNEL command, a terminal response will be sent and parameters will be cleaned up. If the failure was in context with a SEND DATA command, the SIM BIP channel will be closed and parameters will be cleaned up. If the failure was unexpected, a channel status event message will be sent, if enabled.

10.3.2.14 cmhSAT_OpChnGPRSPend

Prototype:

```
BOOL cmhSAT_OpChnGPRSPend( SHORT cid, UBYTE opchStat );
```

Parameters:

cid	: context identifier
opchStat	: open channel status

Return:

BOOL

Description:

This function is used to check whether a GPRS channel request for the given context identifier and status is pending or not.

10.3.2.15 cmhSAT_OpChnGPRSSStat

Prototype:

```
void cmhSAT_OpChnGPRSSStat(T_SAT_GPRS_CB_STAT stat, UBYTE cause);
```

Parameters:

stat	: represents the current GPRS status
cause	: network cause in case of a failure

Return:

none

Description:

This function is called, when the GPRS status has changed. Depending on the GPRS status the SIM channel will be connected, released, suspended or resumed. In case of a problem a channel failure will be indicated and processed accordingly.

10.3.2.16 cmhSAT_OpChnOpen

Prototype:

```
void cmhSAT_OpChnOpen( UBYTE dtiConn, UBYTE chnId );
```

Parameters:

dtiConn	: SIM channel status
chnId	: channel identifier

Return:

none

Description:

This callback function will be called, when the SIM entity has tried to open a SIM channel. In context with the OPEN CHANNEL command and an on demand channel establishment, the command will be terminated and an appropriate result sent to SAT. In case of an unsuccessful channel opening, the command parameters will be cleared.

10.3.2.17 cmhSAT_OpChnSetPPP

Prototype:

```
void cmhSAT_OpChnSetPPP( UBYTE chnType );
```

Parameters:

chnType : channel type (CSD/GPRS)

Return:

none

Description:

This function will set the PPP parameters for CSD channels, if available.

10.3.2.18 cmhSAT_OpChnSIMCnctReq

Prototype:

```
void cmhSAT_OpChnSIMCnctReq(UBYTE unit);
```

Parameters:

unit : data entity to be connected with SIM

Return:

none

Description:

This function is used to request a SIM connection with the given data entity (unit). All necessary parameters are gathered and sent to SIM.

10.3.2.19 cmhSAT_OpChnSIMFail

Prototype:

```
void cmhSAT_OpChnSIMFail( UBYTE dtiConn, UBYTE chnId );
```

Parameters:

dtiConn : SIM channel status
chnId : channel identifier

Return:

none

Description:

This function will be called, if the SIM entity encounters a fail condition with a SIM channel. If the channel is still open, it will be closed immediately. If not and the bearer for that channel is still in use, it will be released immediately.

10.3.2.20 cmhSAT_OpChnStatEvt

Prototype:

```
void cmhSAT_OpChnStatEvt( void );
```

Parameters:

none

Return:

none

Description:

This function will send the CHANNEL STATUS event to SAT.

10.3.2.21 cmhSAT_OpChnUDPActiveCsd

Prototype:

```
void cmhSAT_OpChnUDPActiveCsd(T_ACI_RETURN result);
```

Parameters:

result : outcome of the UDP activation

Return:

none

Description:

This callback function will be called after the UDP stack has been activated for a CSD channel. If the activation was successful the RA entity will be activated next. If not a failure during the open channel process will be indicated.

10.3.2.22 cmhSAT_OpChnUDPActivateGprs

Prototype:

```
void cmhSAT_OpChnUDPActivateGprs( void );
```

Parameters:

none

Return:

none

Description:

This function is used to activate the UDP stack for a GPRS channel request.

10.3.2.23 cmhSAT_OpChnUDPConfCsd

Prototype:

```
void cmhSAT_OpChnUDPConfCsd(T_ACI_RETURN result);
```

Parameters:

result : outcome of the UDP configuration

Return:

none

Description:

This callback function will be called after the UDP stack has been configured for a CSD channel. The next step is the connection of the SIM entity.

10.3.2.24 cmhSAT_OpChnUDPConfGprs

Prototype:

```
void cmhSAT_OpChnUDPConfGprs( void );
```

Parameters:

none

Return:

none

Description:

This callback function is called, when the UDP stack has been configured. The next step is to establish a SIM connection.

10.3.2.25 cmhSAT_OpChnUDPDeactCsd

Prototype:

```
void cmhSAT_OpChnUDPDeactCsd(T_ACI_RETURN result);
```

Parameters:

result : outcome of the UDP deactivation

Return:

none

Description:

This callback function will be called after the UDP stack has been deactivated for a CSD channel. The function has to decide whether the deactivation was intended or unexpected. In the latter case the L2R entity will be deactivated too, and a failure for the SAT data channel will be indicated.

10.3.2.26 cmhSAT_OpChnUDPDeactGprs

Prototype:

```
void cmhSAT_OpChnUDPDeactGprs( void );
```

Parameters:

none

Return:

none

Description:

This callback function is called, when the UDP stack has been deactivated. If the deactivation was unexpected a channel failure will be indicated.

10.3.2.27 cmhSAT_SendData

Prototype:

```
BOOL cmhSAT_SendData( void );
```

Parameters:

none

Return:

BOOL

Description:

This function will be called after reception of a SEND DATA command. The user will be notified about the reception of the command and then the function will check for the conditions of an immediate channel establishment. If that condition applies, a last check for temporary problems will be performed. If successful the channel will be established accordingly to the channel parameters received with the former OPEN CHANNEL command. The function will return TRUE if the command will be executed.

10.3.2.28 cmhSAT_storeCSDPrms

Prototype:

```
BOOL cmhSAT_storeCSDPrms( T_OPEN_CHANNEL *opchCmd );
```

Parameters:

opchCmd : pointer to the decoded open channel command parameters

Return:

BOOL

Description:

This function will allocate memory to store all command parameters that are necessary to establish the CSD channel after the user has accepted the pending channel request.

10.3.2.29 cmhSAT_storeGPRSPrms

Prototype:

```
BOOL cmhSAT_storeGPRSPrms( T_OPEN_CHANNEL *opchCmd );
```

Parameters:

opchCmd : pointer to the decoded open channel command parameters

Return:

BOOL

Description:

This function will allocate memory to store all command parameters that are necessary to establish the GPRS channel after the user has accepted the pending channel request.

10.3.2.30 cmhSAT_UserAcptCntxt

Prototype:

```
BOOL cmhSAT_UserAcptCntxt( void );
```

Parameters:

none

Return:

BOOL

Description:

This function will be called if the user accepts a pending context activation requested by SAT to open a GPRS data channel. In case of a on demand channel establishment, the function will only request the SIM entity to open a data channel. In case of immediate channel establishment, a last check for temporary problems will be performed. If successful the context will be activated accordingly to the channel parameters received with the OPEN CHANNEL command. The function will return TRUE if the context will be established. If an error has occurred and the context will not be established, the function will return FALSE.

10.3.2.31 cmhSAT_UserRejCntxt

Prototype:

```
void cmhSAT_UserRejCntxt( void );
```

Parameters:

none

Return:

none

Description:

This function will be called if the user rejects a pending context activation requested by SAT to open a GPRS data channel. The function will then generate the appropriate terminal response for the OPEN CHANNEL command.

10.3.3 Module cmh_smf.c

10.3.3.1 cmhSM_getCurQOS

Prototype:

```
T_QOS* cmhSM_getCurQOS( SHORT cid );
```

Parameters:

cid : context id

Return:

T_QOS

Description:

The function returns a pointer to the quality of service parameters of the context identified by the cid parameter.

10.3.4 Module dti_manager.c

10.3.4.1 check_ind_dti_id

Prototype:

```
BOOL check_int_dti_id(UBYTE dti_id);
```

Parameters:

dti_id : dti channel identifier

Return:

none

Description:

This function checks if dti_id has been created internal.

10.3.4.2 SIM_ENT_CSDconnect_dti_cb

Prototype:

```
BOOL SIM_ENT_CSDconnect_dti_cb(UBYTE dti_id, T_DTI_MNG_CB_TYPE result_type);
```


Parameters:

dti_id : dti channel identifier
result_type : result of dti channel establishment

Return:

TRUE

Description:

This is the callback function of dti manager for the connection between SIM and L2R/TRA or UDP entity. In error case the disconnection of entities are started.

10.3.4.3 SIM_ENT_GPRSconnect_dti_cb

Prototype:

BOOL SIM_ENT_GPRSconnect_dti_cb(UBYTE dti_id, T_DTI_MNG_CB_TYPE result_type);

Parameters:

dti_id : dti channel identifier
result_type : result of dti channel establishment

Return:

TRUE

Description:

This is the callback function of dti manager for the connection between SIM and SNDCP/UDP entity. In error case the disconnection of entities are started.

10.3.4.4 SIM_SNDCP_connect_dti_cb

Prototype:

BOOL SIM_SNDCP_connect_dti_cb(UBYTE dti_id, T_DTI_MNG_CB_TYPE result_type);

Parameters:

dti_id : dti channel identifier
result_type : result of dti channel establishment

Return:

TRUE

Description:

This is the callback function of dti manager for the connection between SIM and SNDCP entity. In error case the disconnection of entities are started.

10.3.5 Module psa_satf.c

10.3.5.1 psaSAT_cnvrtAPN2NetworkAdr

Prototype:

SHORT psaSAT_ctbFindActCall(void);

Parameters:

none

Return:

SHORT

Description:

This function will search for a call in an active state listed in the call table. The function returns the call identifier of the call or NO_ENTRY if no call could be found matching the search criteria.

10.3.5.2 psaSAT_gprsFindFreeCntxt

Prototype:

SHORT psaSAT_gprsFindFreeCntxt(void);

Parameters:

none

Return:

SHORT

Description:

This function will search for a free PDP context identifier. The function returns the context identifier for the available context or NO_ENTRY if no free context could be found.

10.3.6 Module psa_simp.c

10.3.6.1 psa_sim_dti_cnf

Prototype:

```
void psa_sim_dti_cnf(T_SIM_DTI_CNF *sim_dti_cnf);
```

Parameters:

sim_dti_cnf : points to the primitive SIM_DTI_CNF

Return:

none

Description:

This function evaluates the content of the primitive SIM_DTI_CNF sent by SIM entity. The results are sent via callback function to inform the MMI. The structure T_SIM_DTI_CH_PRM are freed.

10.3.6.2 psa_sim_dti_ind

Prototype:

```
void psa_sim_dti_ind ( T_SIM_DTI_IND *sim_dti_ind );
```

Parameters:

sim_dti_ind : points to the primitive SIM_DTI_IND

Return:

none

Description:

This function receives the primitive SIM_DTI_IND sent by SIM. The primitive indicates closing of BIP channel and dti connection if the release timer is expired.

10.3.6.3 psa_sim_eventlist_cnf

Prototype:

```
void psa_sim_eventlist_cnf ( T_SIM_EVENTLIST_CNF *sim_eventlist_cnf );
```

Parameters:

sim_eventlist_cnf : points to the primitive SIM_EVENTLIST_CNF

Return:

none

Description:

This function handles the primitive SIM_EVENTLIST_CNF sent by SIM entity. The parameter event_data_avail is checked.

10.3.7 Module psa_sims.c

10.3.7.1 psaSIM_Dti_Req

Prototype:

```
void psaSIM_Dti_Req;
```

Parameters:

none

Return:

none

Description:

This function sends the primitive SIM_DTI_REQ to the SIM entity. The sent parameters depend on the state of connection (open channel, close channel, resume channel, suspended channel), on the kind of bearer (CSD or GPRS) and on the kind of partner entity (UDP, TRA, L2R, SNDCP). The SIM entity receives all needed parameters for the channel establishment resp. release which are sent in the OPEN_CHANNEL resp. CLOSE_CHANNEL commands.

10.3.7.2 psaSIM_EvDatAvail

Prototype:

```
void psaSIM_EvDatAvail ( BOOL evStat );
```

Parameters:

evStat : channel parameter

Return:

none

Description:

This function sends the primitive SIM_EVENTLIST_REQ to the SIM entity. The SIM entity is informed about a change of the status of the Data available event

10.3.7.3 psaSIM_SATChn

Prototype:

```
void psaSIM_SATChn ( T_SIM_SAT_CHN chnInf, void (*cb)(UBYTE dtiConn, UBYTE chnId);
```

Parameters:

chnInf : channel parameter
cb : callback function for results of DTI connection
dtiConn : SIM channel status
chnId : channel identifier

Return:

none

Description:

This function starts the dti manager for the connection resp. disconnection between SIM entity and a partner entity like UDP, L2R, TRA, SNDCP. The delivered channel parameter and the callback function are stored in the allocated structure T_SIM_DTI_CH_PRM for the later use. The function acts in different way with respect to the kind of bearer and channel status (case on demand or immediately) and on kind of partner entity.

10.3.8 Module psa_udpas.c

0.1.1.10 psaUDPA_Dti_Req

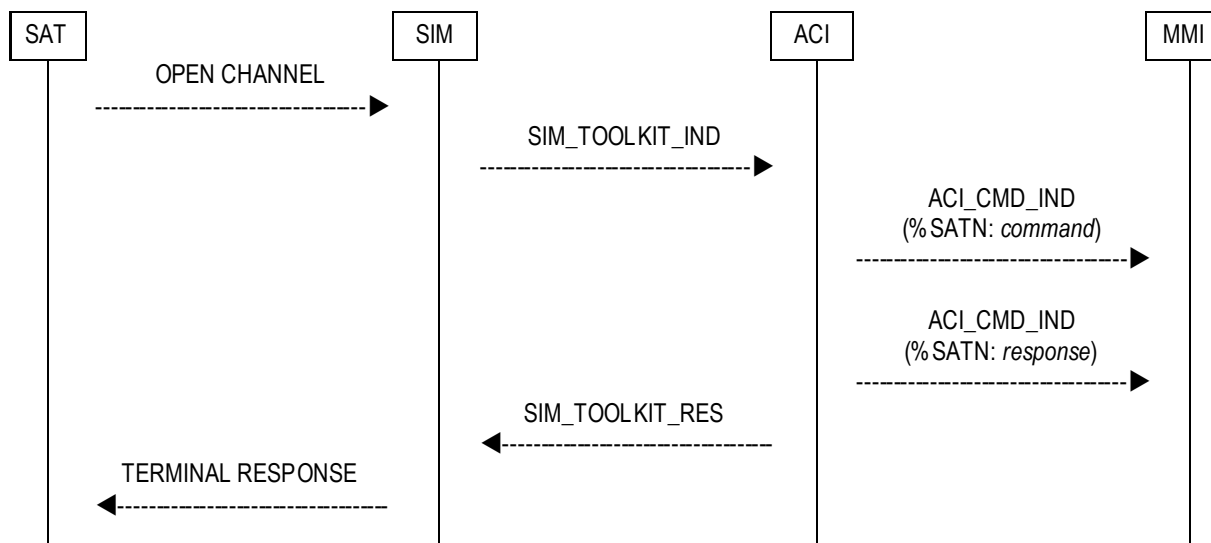
- send primitive UDPA_CONFIG_REQ with parameters entity_name, tui, layer, c_id and dti_conn
- inform entity UDP about connection with entity SIM on the upper layer

11 MSC

11.1 OPEN CHANNEL, confirmation phase

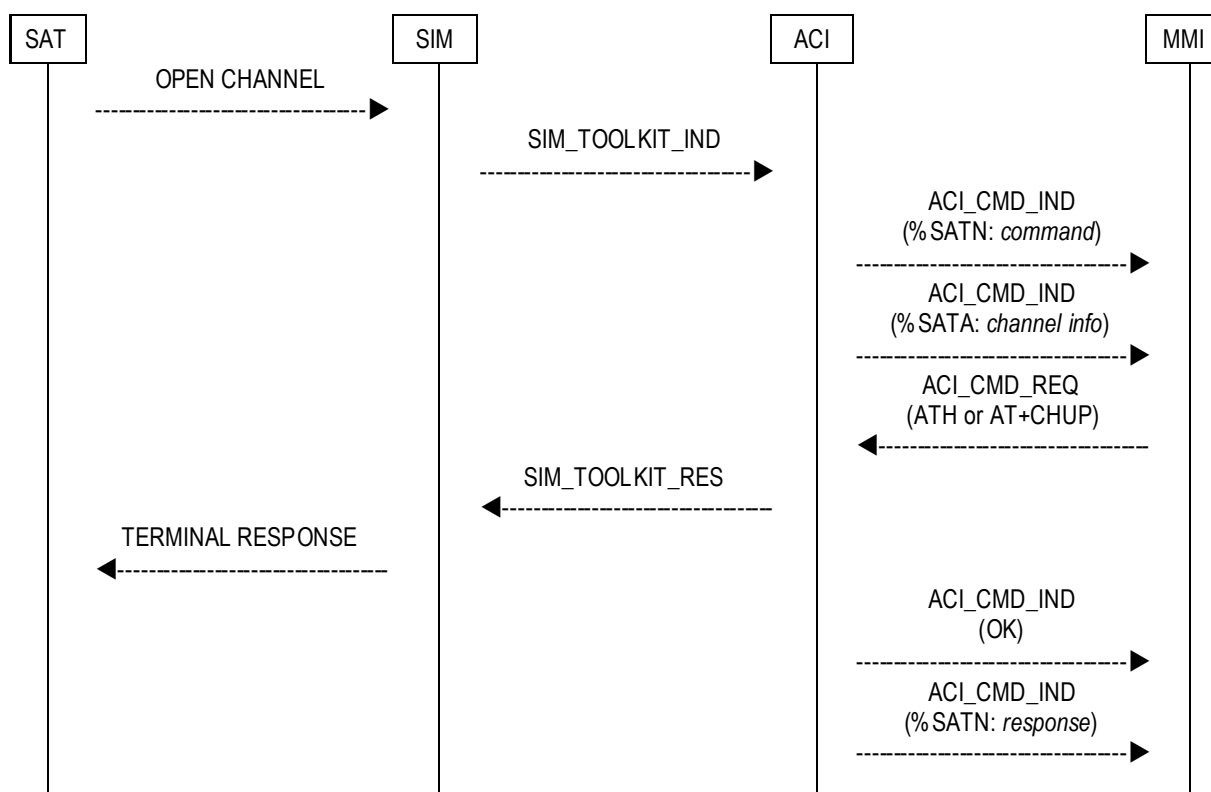
11.1.1 OPEN CHANNEL, command will be rejected because of unsupported parameters

After reception of the OPEN CHANNEL command, ACI will check for basic conditions that have to be fulfilled. If command parameters are not supported by ACI (e.g. IPv6 address format) the command will be rejected immediately with the cause "Command beyond ME's capabilities".



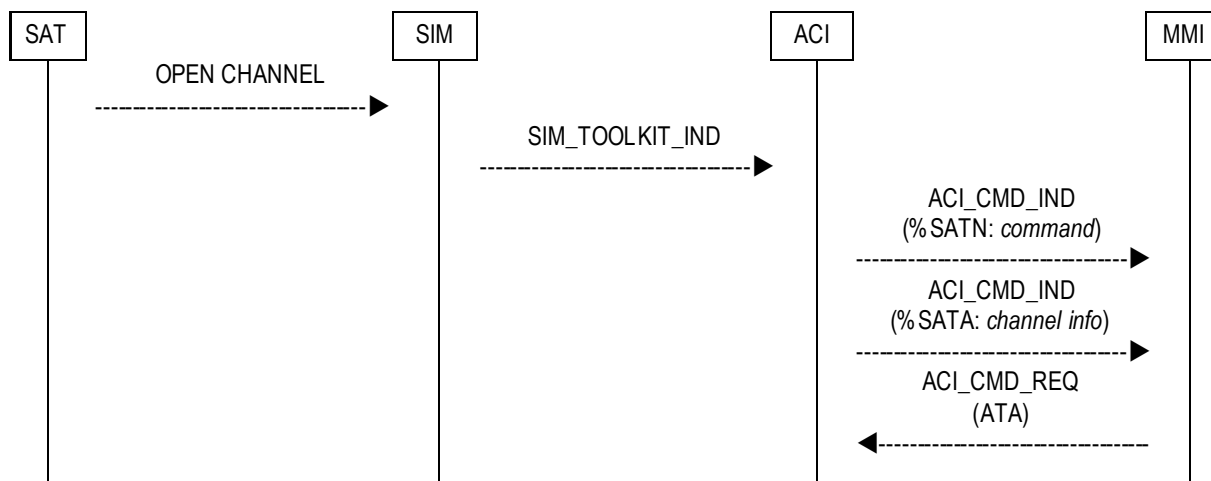
11.1.2 OPEN CHANNEL, user rejects open channel establishment attempt

ACI receives an OPEN CHANNEL command. After checking the command no conditions apply to reject the command. Therefore the user will be notified about the pending channel request by SAT. The user rejects the channel request by sending the AT command ATH or ST+CHUP. The command will be terminated by ACI with the cause "User did not accept the proactive command"



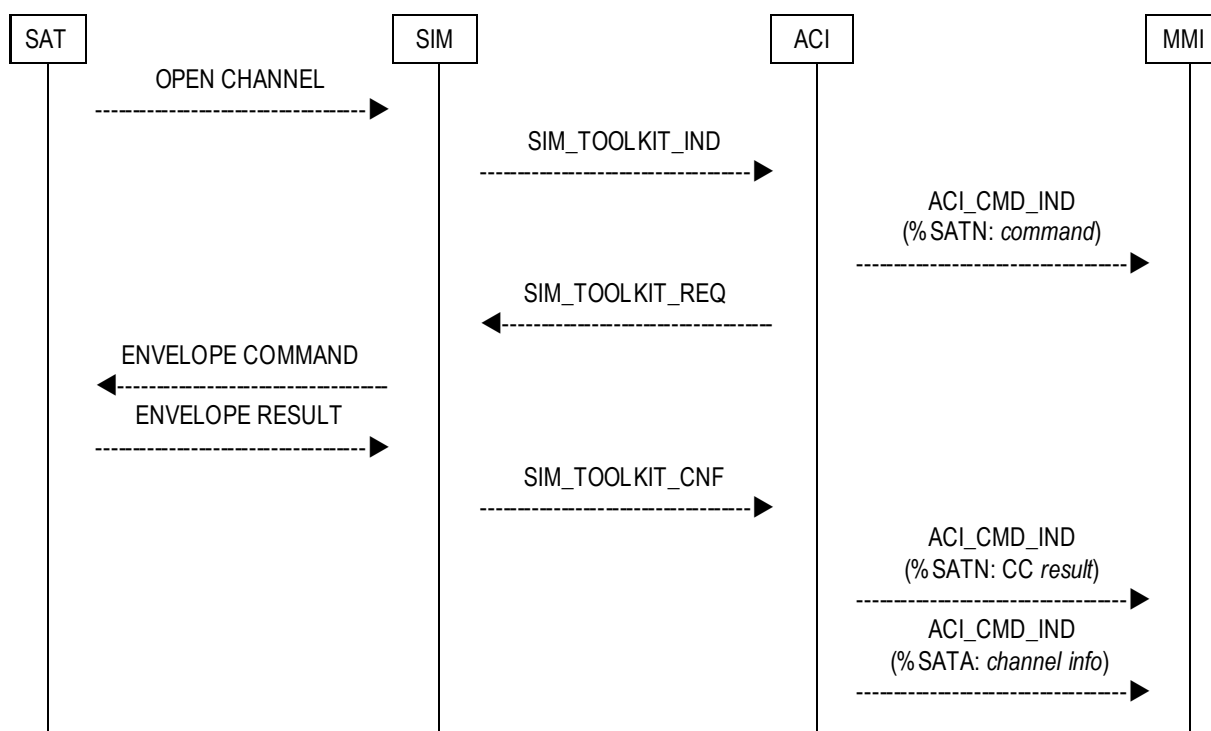
11.1.3 OPEN CHANNEL, user accepts open channel establishment attempt

ACI receives an OPEN CHANNEL command. After checking the command no conditions apply to reject the command. Therefore the user will be notified about the pending channel request by SAT. The user accepts the channel request by sending the AT command ATA. The command will be processed accordingly to the command parameters. See the following MCS's in the next chapter (11.2)



11.1.4 OPEN CHANNEL, channel establishment allowed by Call Control by SIM

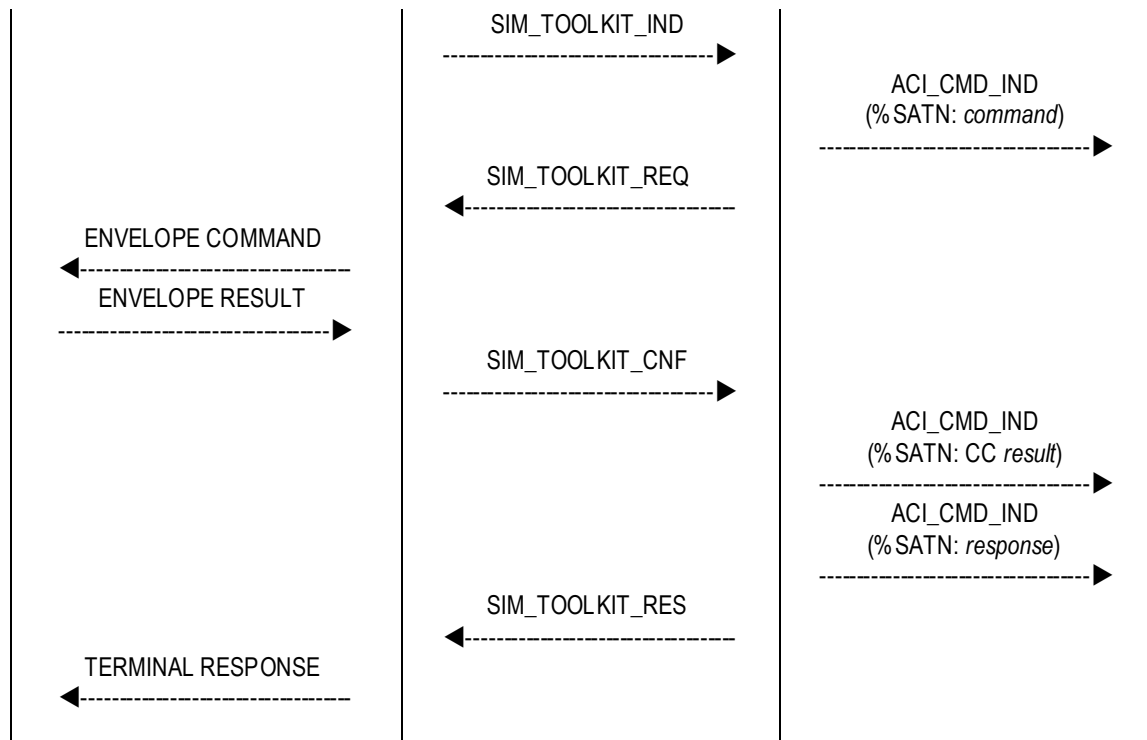
Call Control by SIM is allocated and activated for the SIM card. After reception of an OPEN CHANNEL command, ACI has to pass the set up parameters for CSD channels to SIM. SIM accepts the request and ACI informs the user about the pending channel request. Now the user can accept or reject the channel establishment.



11.1.5 OPEN CHANNEL, channel establishment not allowed by Call Control by SIM

Call Control by SIM is allocated and activated for the SIM card. After reception of an OPEN CHANNEL command, ACI has to pass the set up parameters for CSD channels to SIM. SIM rejects the request and ACI will terminate the command with the cause "Interaction with call control by SIM or MO short message control by SIM, permanent problem".

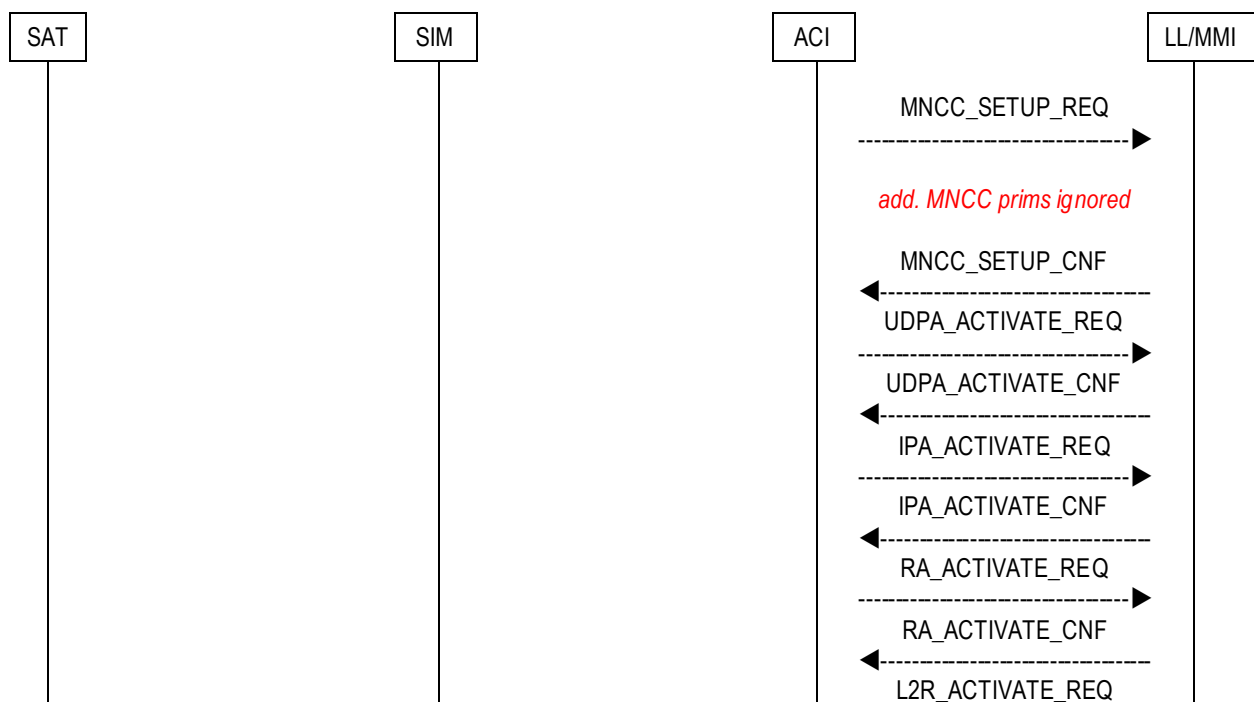


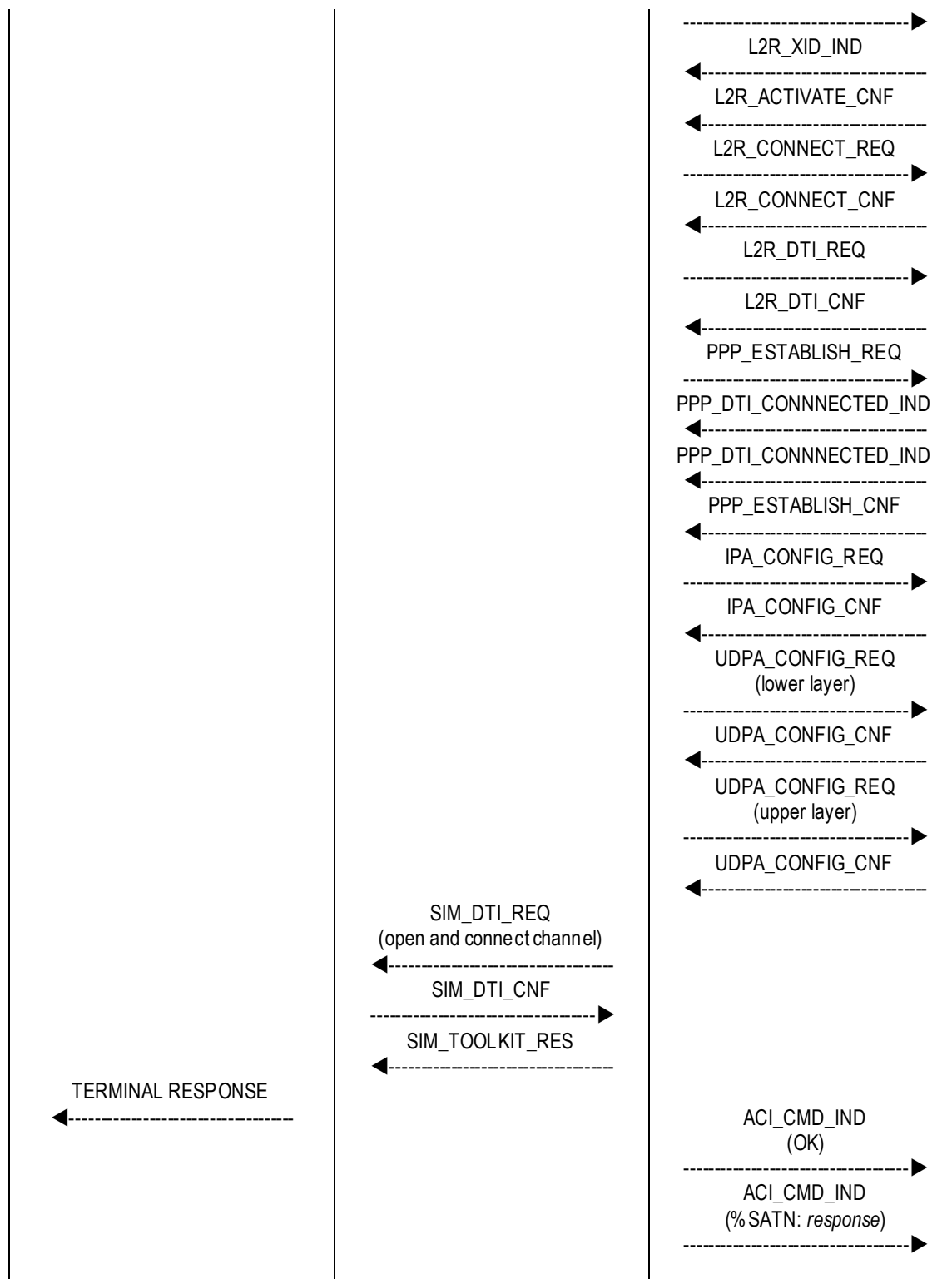


11.2 OPEN CHANNEL, establishment phase

11.2.1 OPEN CHANNEL, immediate channel establishment, CSD, UDP

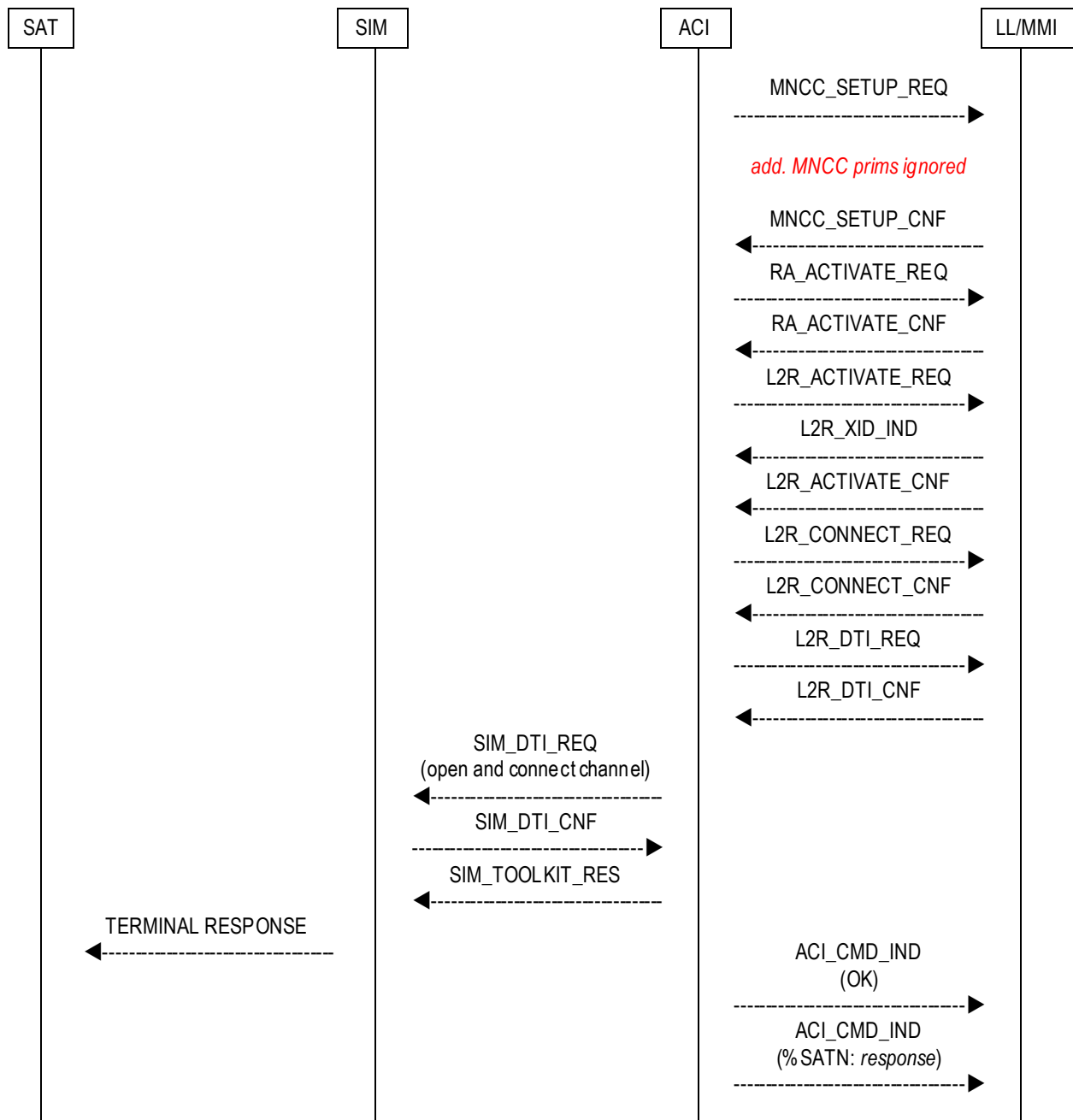
See MSC (11.1.3) as a predecessor for this scenario. SAT has requested an immediate channel establishment for a CSD connection with UDP as the interface transport level. After acceptance by the user ACI proceeds with channel establishment. If the network connection has been established and the data chain is set up, the command will be terminated with the cause "Command performed successfully"





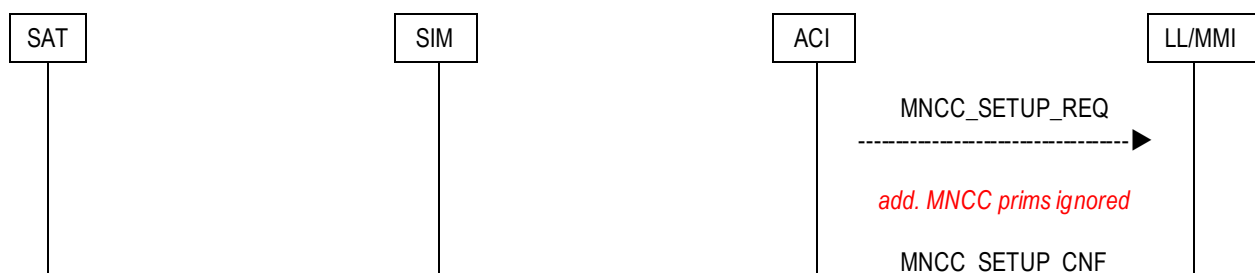
11.2.2 OPEN CHANNEL, immediate channel establishment, CSD, non-transparent

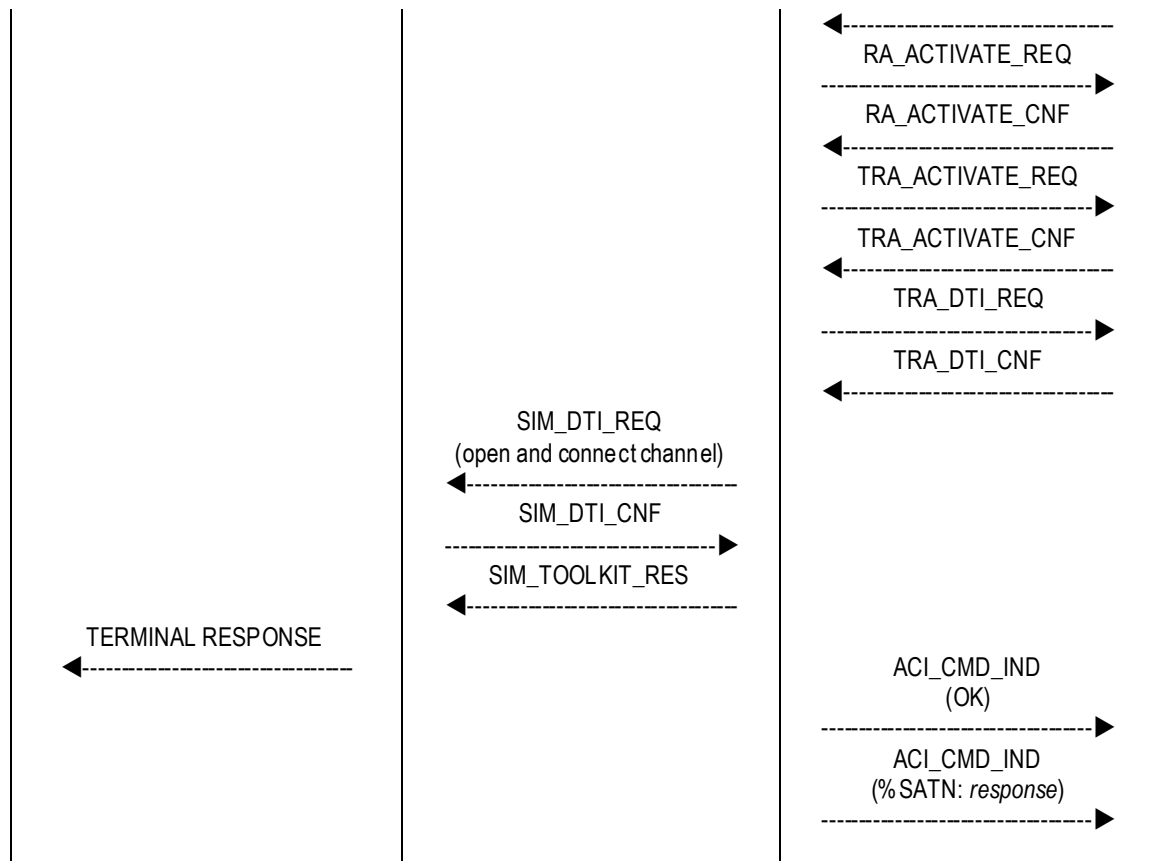
See MSC (11.1.3) as a predecessor for this scenario. SAT has requested an immediate channel establishment for a CSD connection in non-transparent mode. After acceptance by the user ACI proceeds with channel establishment. If the network connection has been established and the data chain is set up, the command will be terminated with the cause "Command performed successfully"



11.2.3 OPEN CHANNEL, immediate channel establishment, CSD, transparent

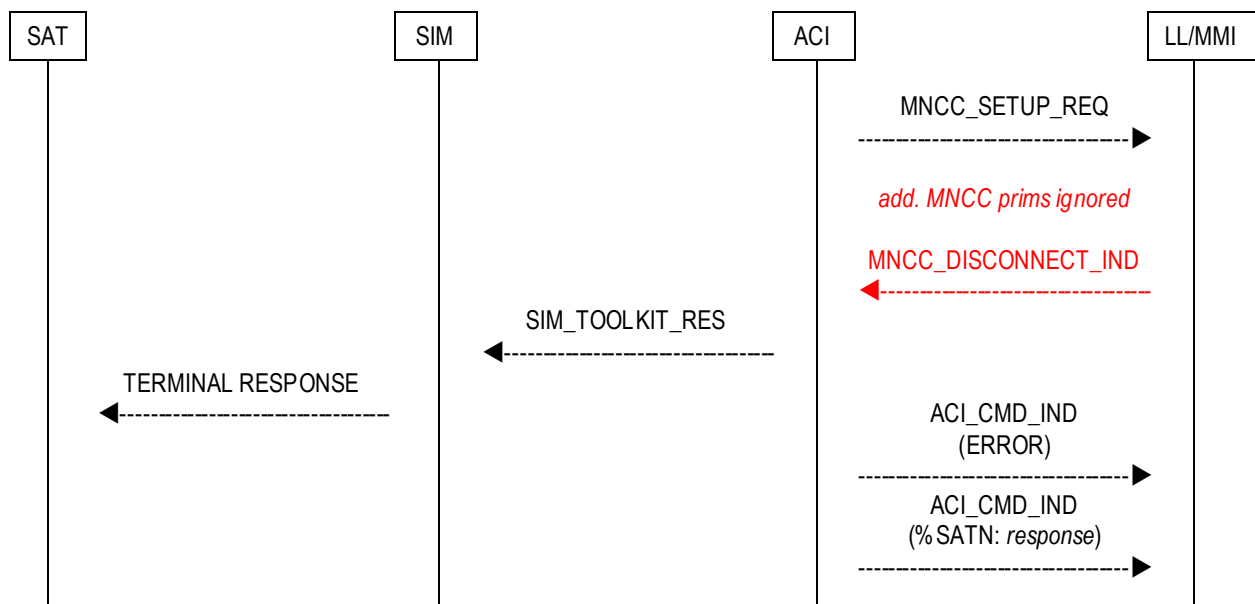
See MSC (11.1.3) as a predecessor for this scenario. SAT has requested an immediate channel establishment for a CSD connection in transparent mode. After acceptance by the user ACI proceeds with channel establishment. If the network connection has been established and the data chain is set up, the command will be terminated with the cause "Command performed successfully"





11.2.4 OPEN CHANNEL, immediate channel establishment, CSD, rejected by the network

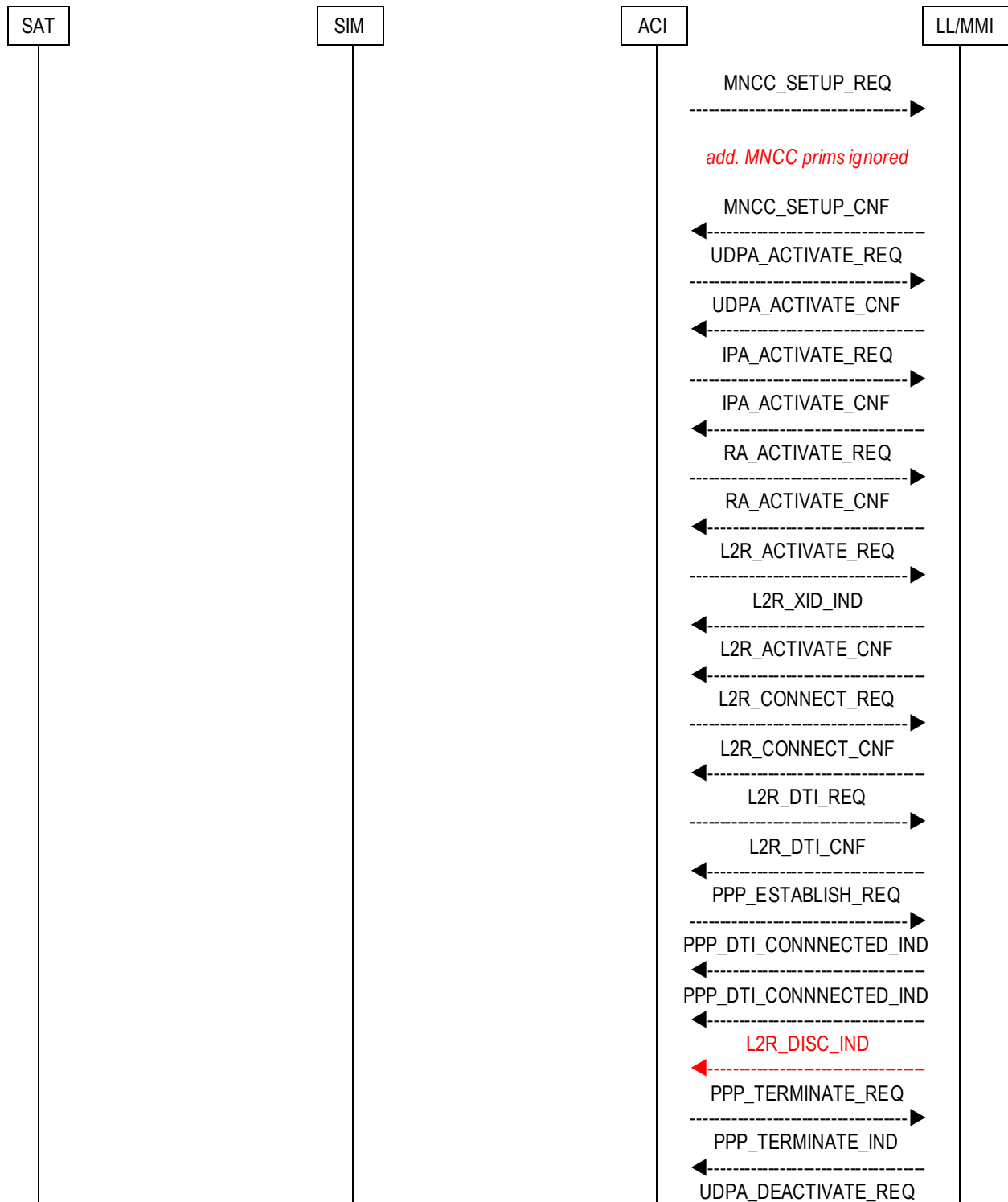
See MSC (11.1.3) as a predecessor for this scenario. SAT has requested an immediate channel establishment for a CSD connection. After acceptance by the user ACI proceeds with channel establishment. In this scenario the network indicates a problem and rejects the call set up. The command will be terminated with the cause "Network currently unable to process command"

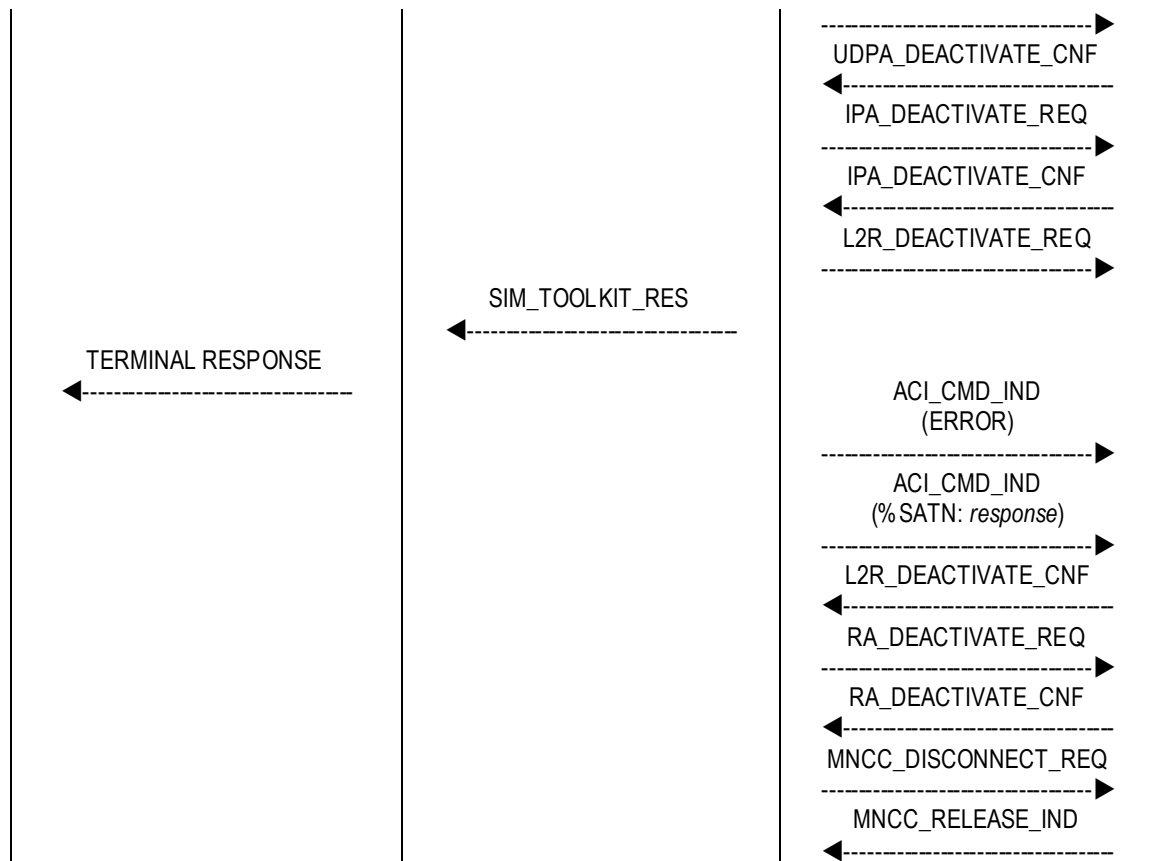


11.2.5 OPEN CHANNEL, immediate channel establishment, CSD, failure within the data chain

See MSC (11.1.3) as a predecessor for this scenario. SAT has requested an immediate channel establishment for a CSD connection with UDP as the interface transport level. After acceptance by the user ACI proceeds with channel establishment. During the establishment of the data chain, the L2R entity detects a fail condition. The channel establishment will be aborted and the connection released. The command will be terminated with the cause "ME currently unable to process command"

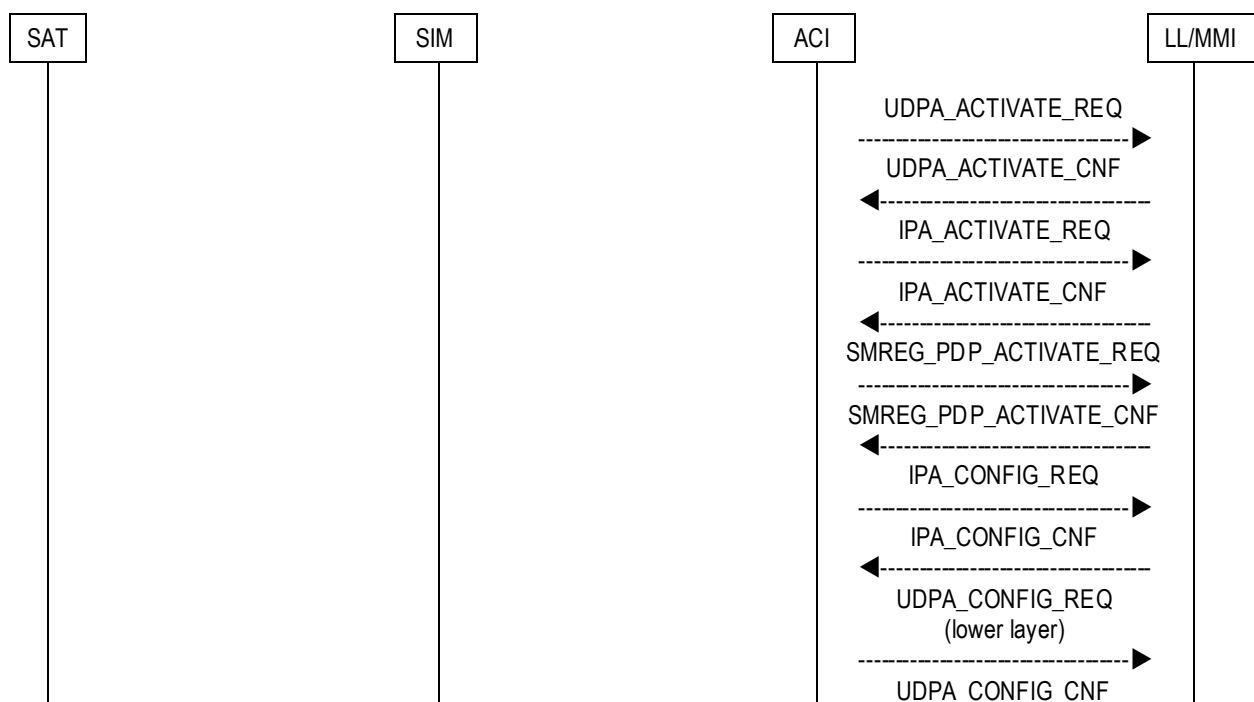
In this scenario the L2R entity is the source of the problem. For other data entities the scenario will be similar with minimal modifications. Therefore not all possible combinations are listed here.

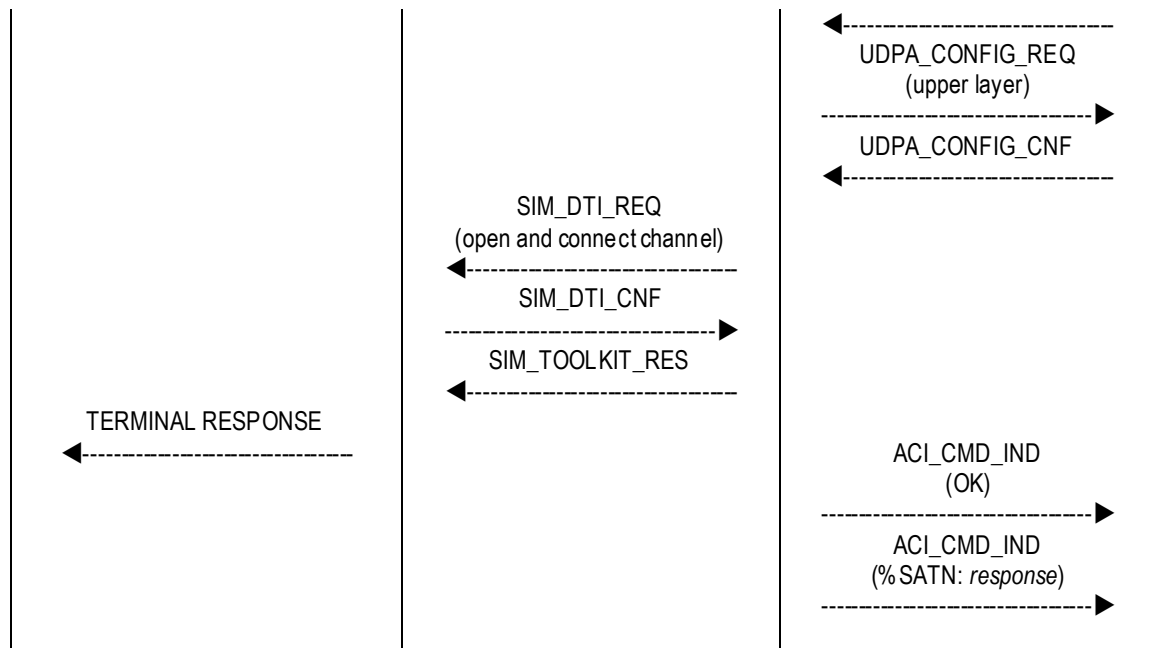




11.2.6 OPEN CHANNEL, immediate channel establishment, GPRS, UDP

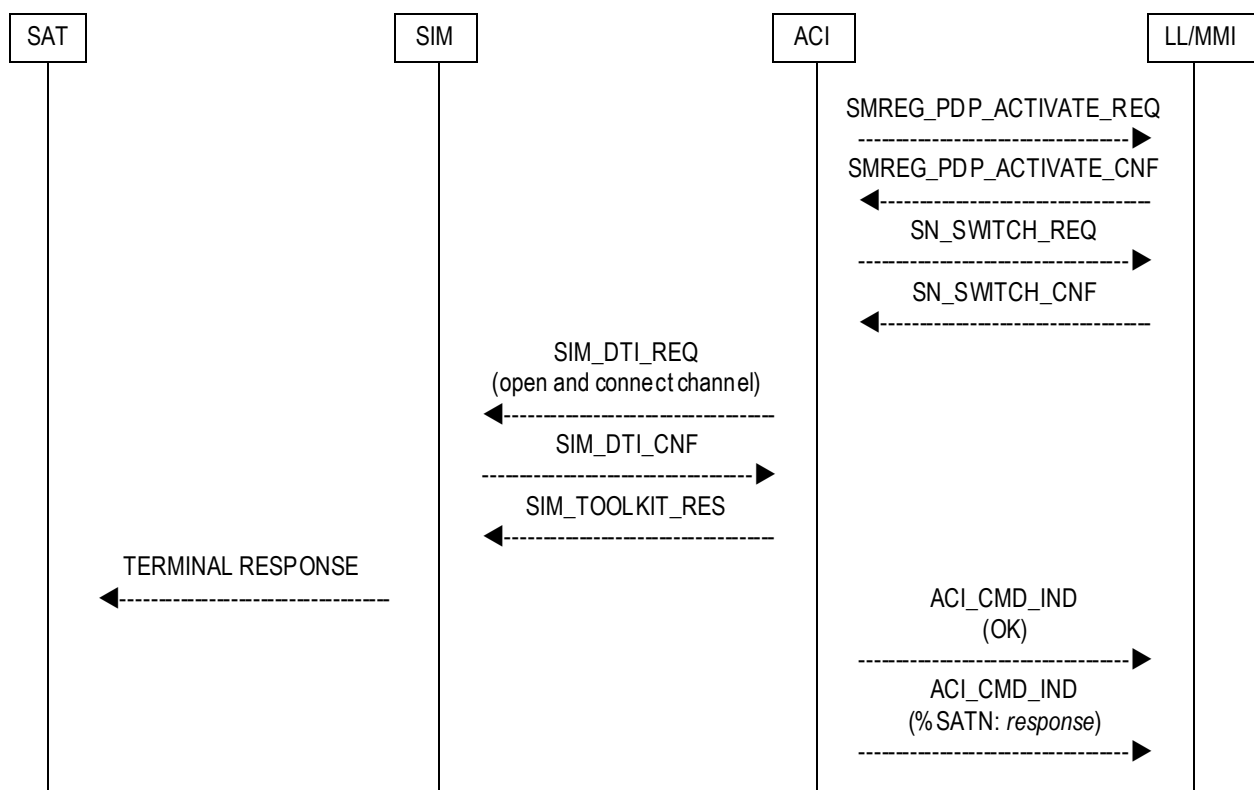
See MSC (11.1.3) as a predecessor for this scenario. SAT has requested an immediate channel establishment for a GPRS connection with UDP as the interface transport level. After acceptance by the user ACI proceeds with channel establishment. If the PDP context has been activated and the data chain is set up, the command will be terminated with the cause "Command performed successfully"





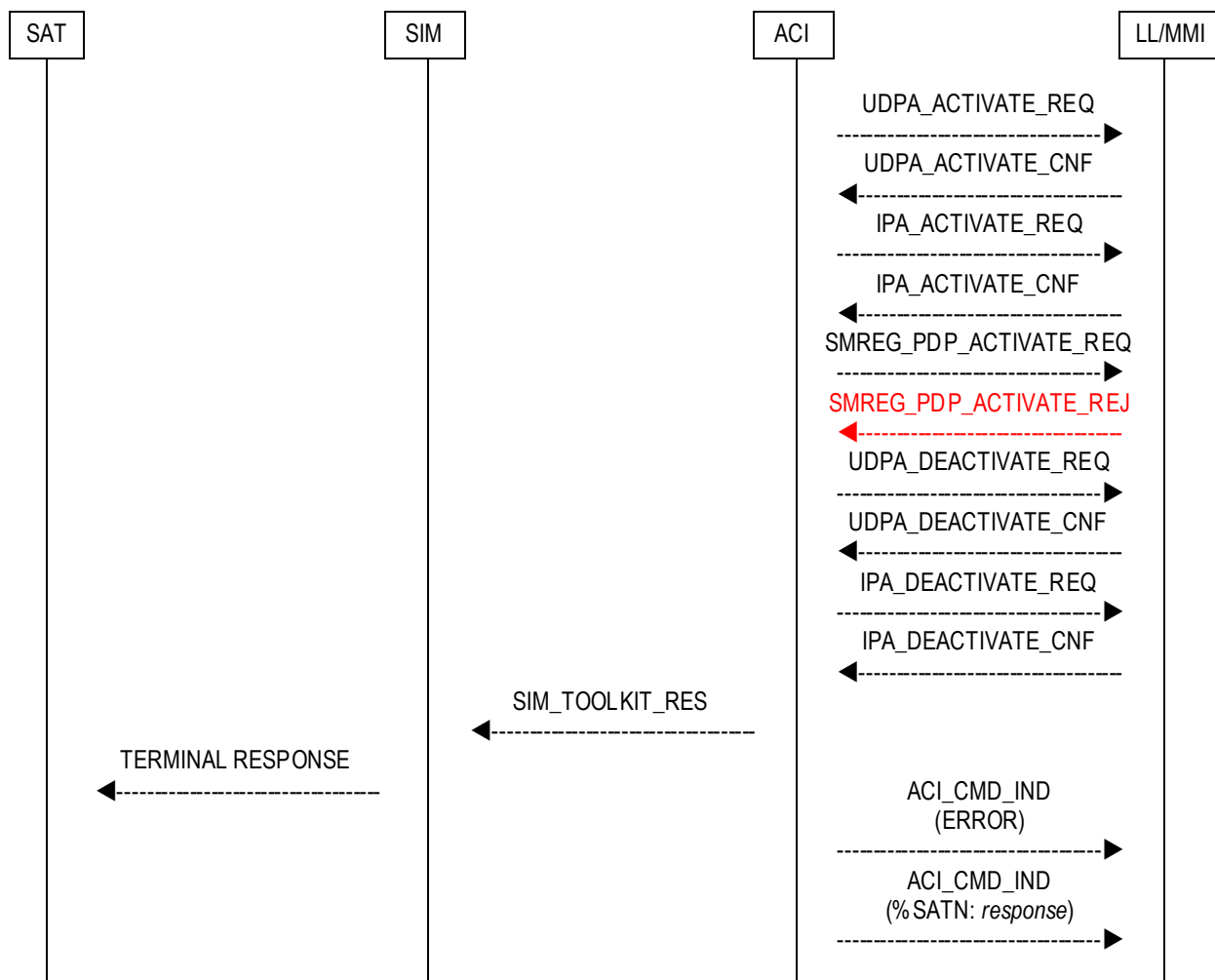
11.2.7 OPEN CHANNEL, immediate channel establishment, GPRS, SNDCP

See MSC (11.1.3) as a predecessor for this scenario. SAT has requested an immediate channel establishment for a GPRS connection with no specific transport layer. After acceptance by the user ACI proceeds with channel establishment. If the PDP context has been activated and the data chain is set up, the command will be terminated with the cause "Command performed successfully"



11.2.8 OPEN CHANNEL, immediate channel establishment, GPRS, rejected by the network

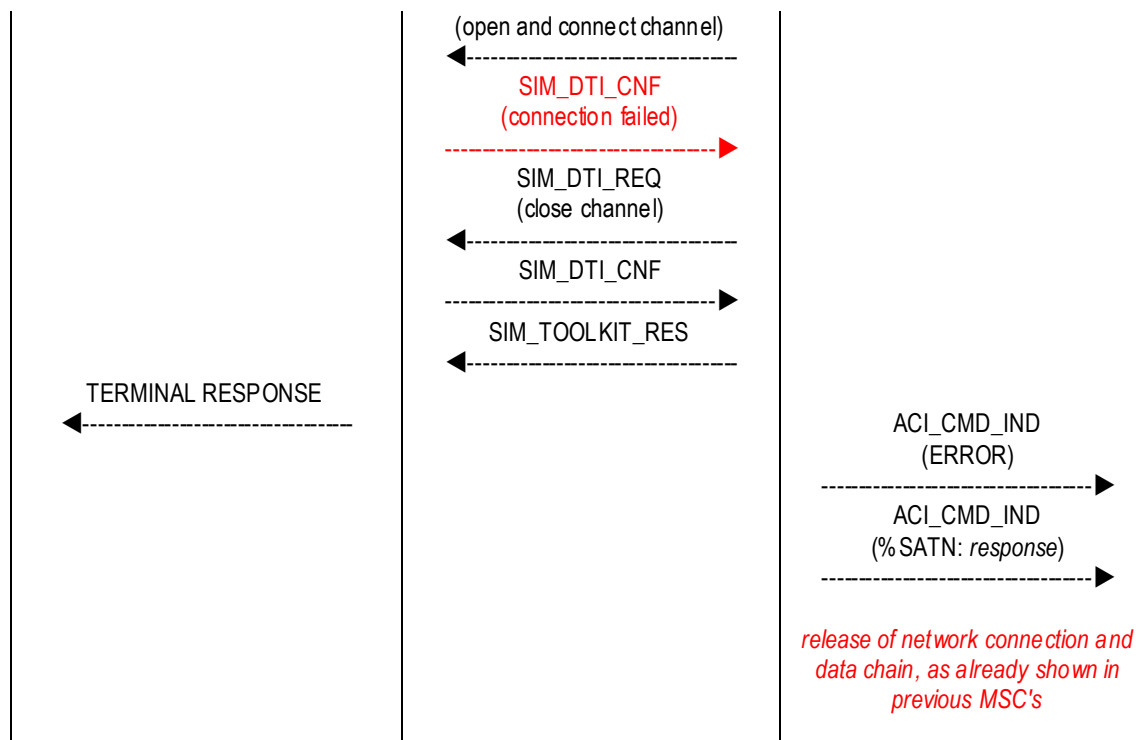
See MSC (11.1.3) as a predecessor for this scenario. SAT has requested an immediate channel establishment for a GPRS connection with UDP as the interface transport level. After acceptance by the user ACI proceeds with channel establishment. The network will reject the PDP context activation. The data chain will be released and the command will be terminated with the cause "Network currently unable to process command"



11.2.9 OPEN CHANNEL, immediate channel establishment, SIM connection failure

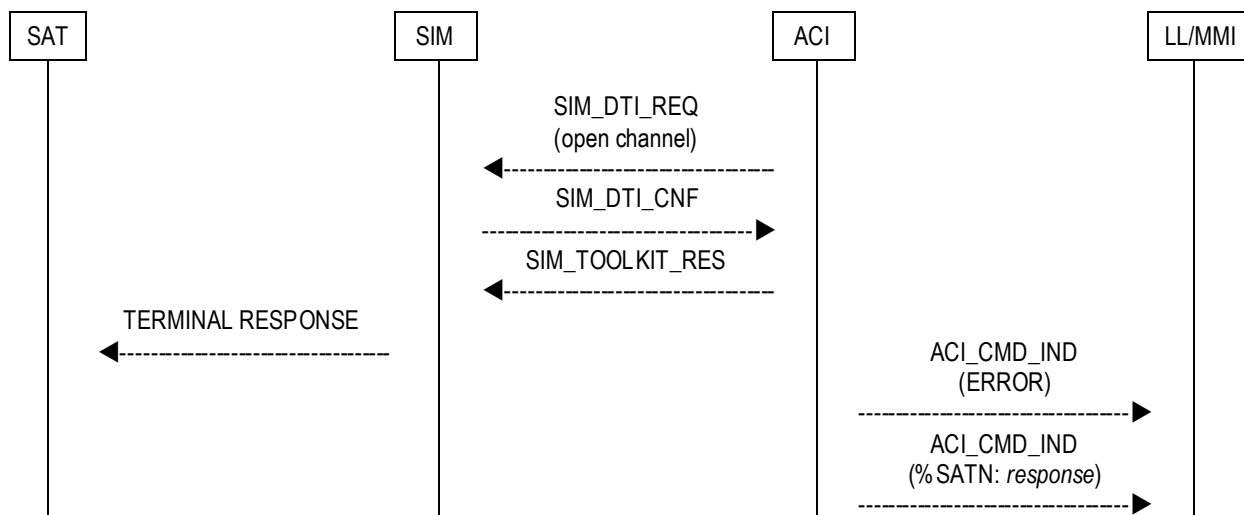
See MSC (11.1.3) as a predecessor for this scenario. SAT has requested an immediate channel establishment for a CSD or GPRS connection. After acceptance by the user ACI proceeds with channel establishment. The network connection has been established and the data chain is set up (this can be seen in the numerous MSC's in this chapter). Only the SIM entity reports a problem and leaves the SIM channel open. Therefore the SIM channel will be closed and the command will be terminated with the cause "ME currently unable to process command"





11.2.10 OPEN CHANNEL, on demand channel establishment

See MSC (11.1.3) as a predecessor for this scenario. SAT has requested an on demand channel establishment for a CSD or GPRS connection. After acceptance by the user ACI will open the SIM channel only and the command will be terminated with the cause "Command performed successfully"



11.3 SEND DATA

11.3.1 SEND DATA, on demand

See MSC (11.2.1) as a predecessor for this scenario. A SAT data channel is up and running. In this case all SEND DATA commands will be passed to MMI, without any action by ACI. The handling of the data transmission will be done by SIM. In case of display information within the SEND DATA command, MMI has to act accordingly.

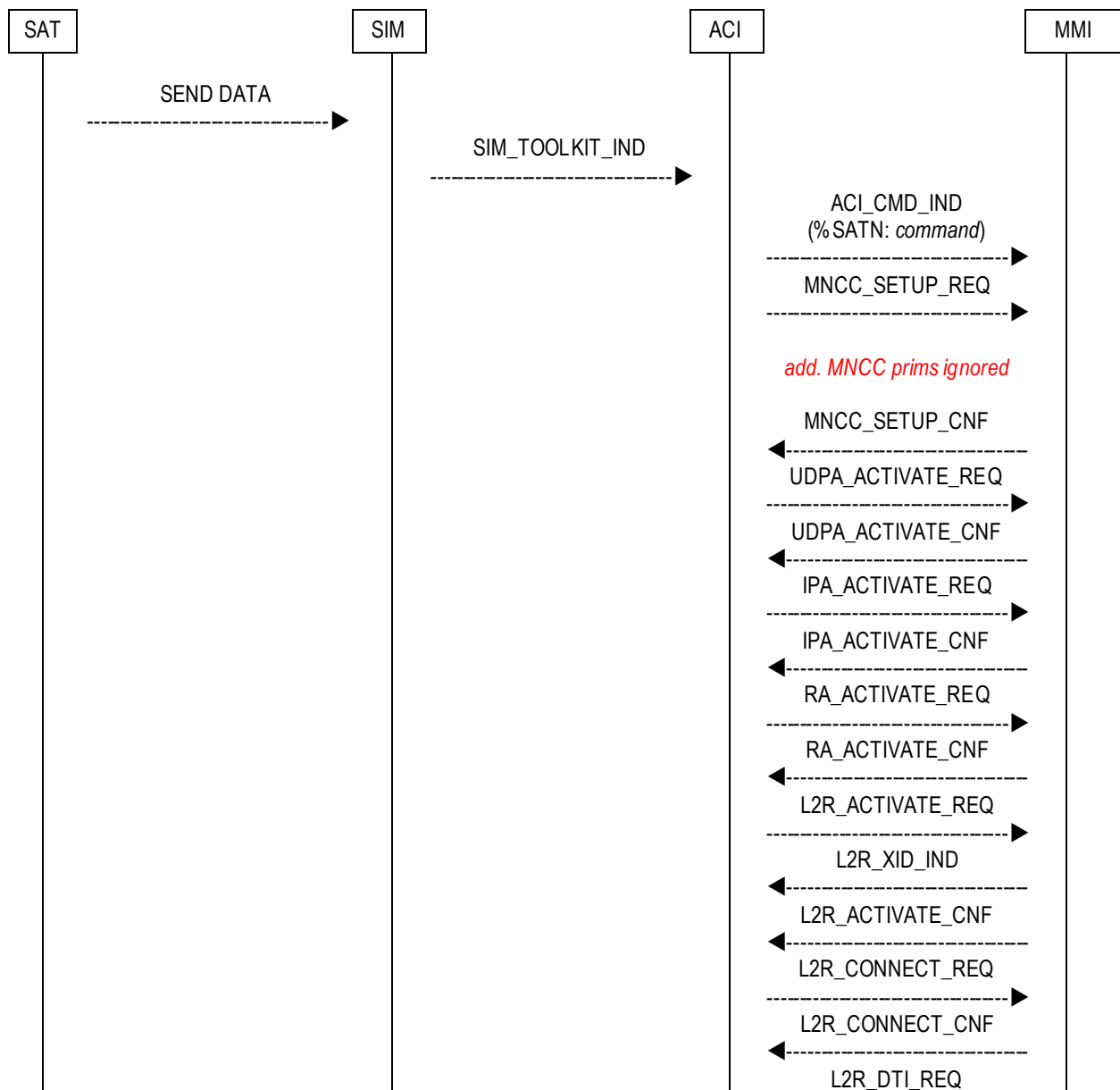


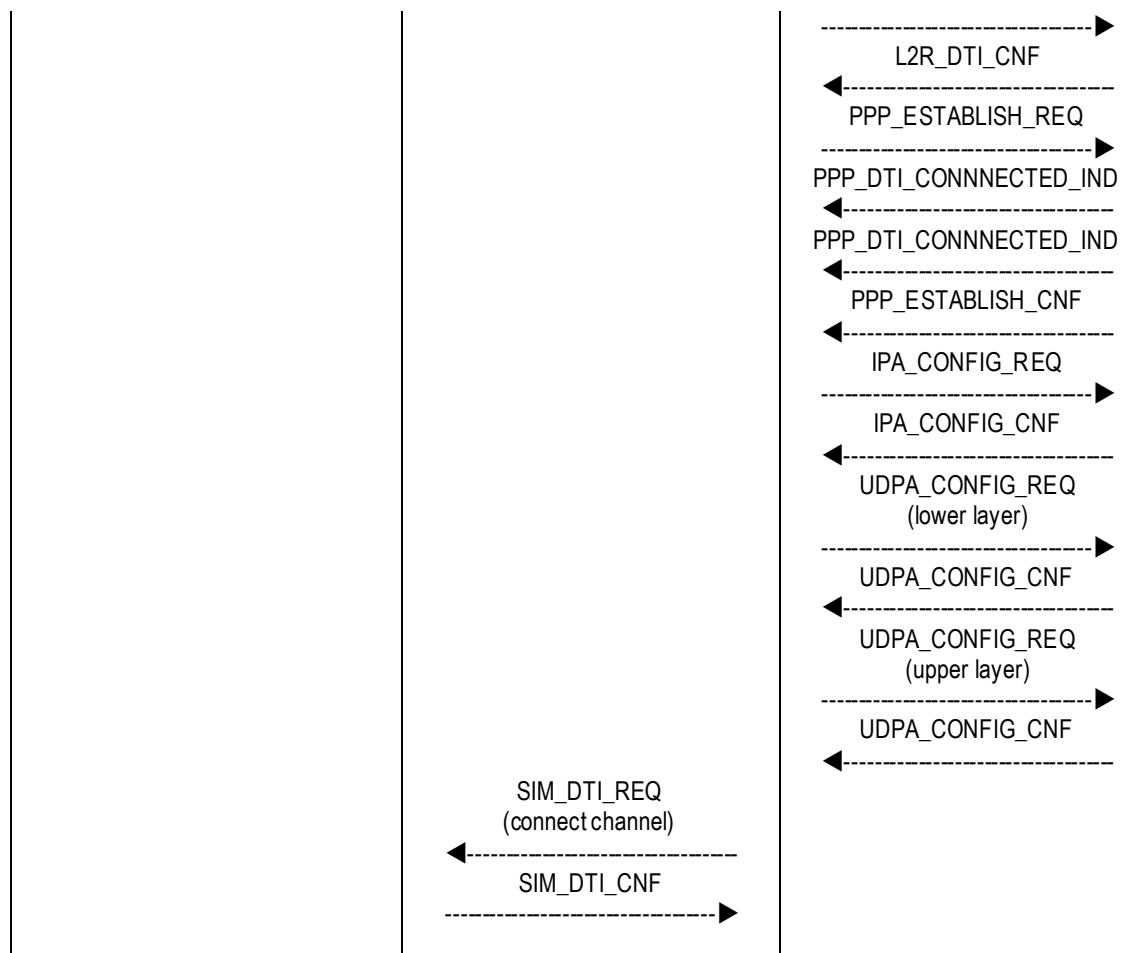


11.3.2 SEND DATA, immediate, CSD, UDP

See MSC (11.2.10) as a predecessor for this scenario. After SAT has requested an on demand channel establishment, the first SEND DATA command with the command details flag set to "immediate" will trigger the channel establishment. For ACI the channel establishment ends with the successful SIM channel connection. SIM is in charge to terminate the command finally.

The scenarios for transparent and non-transparent data channels are similar. The only differences are the data entities that are involved. See the examples in chapter (11.2)

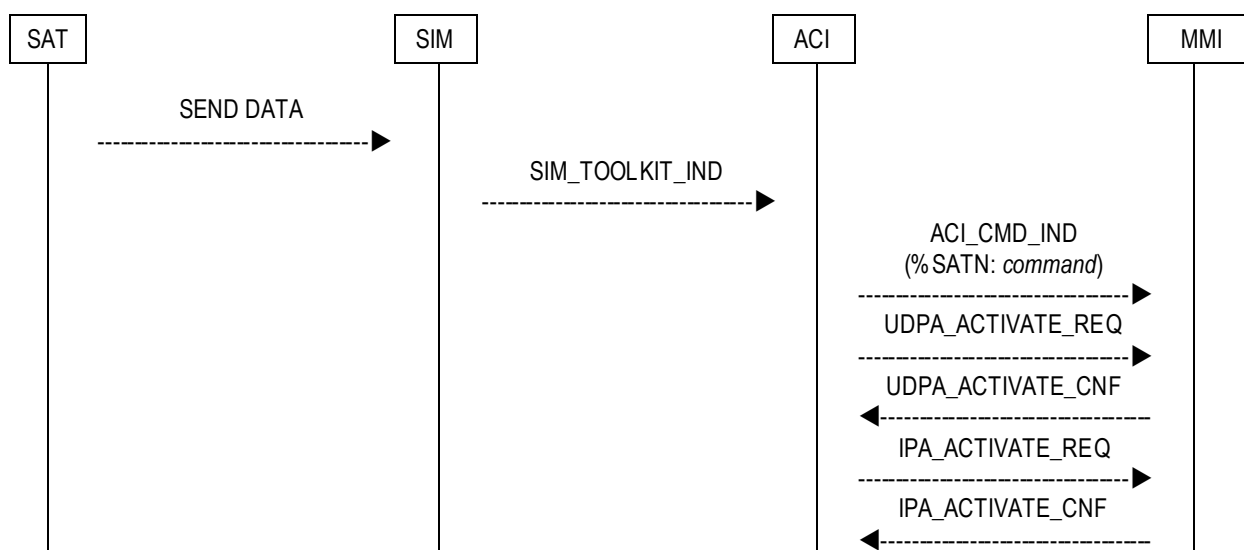


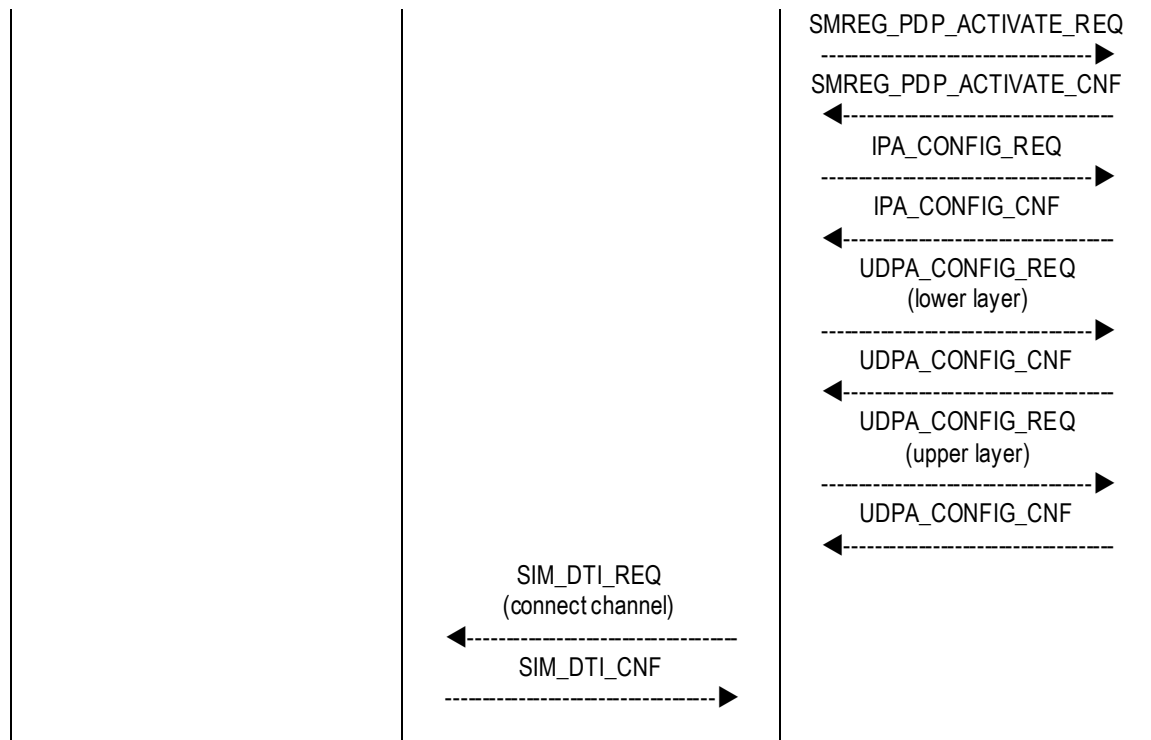


11.3.3 SEND DATA, immediate, GPRS, UDP

See MSC (11.2.10) as a predecessor for this scenario. After SAT has requested an on demand channel establishment, the first **SEND DATA** command with the command details flag set to "immediate" will trigger the channel establishment. For ACI the channel establishment ends with the successful SIM channel connection. SIM is in charge to terminate the command finally.

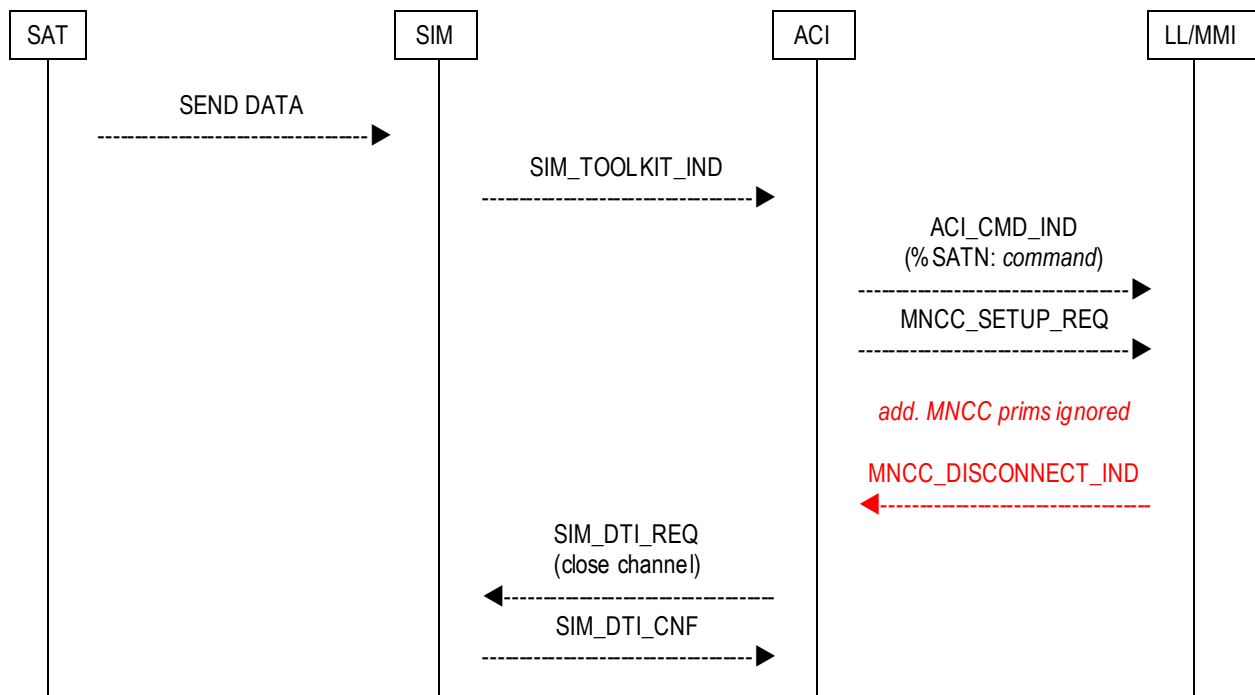
The scenario for SNDTCP only data channels is similar. The only differences are the data entities that are involved. See the examples in chapter (11.2)





11.3.4 SEND DATA, immediate, CSD, rejected by the network

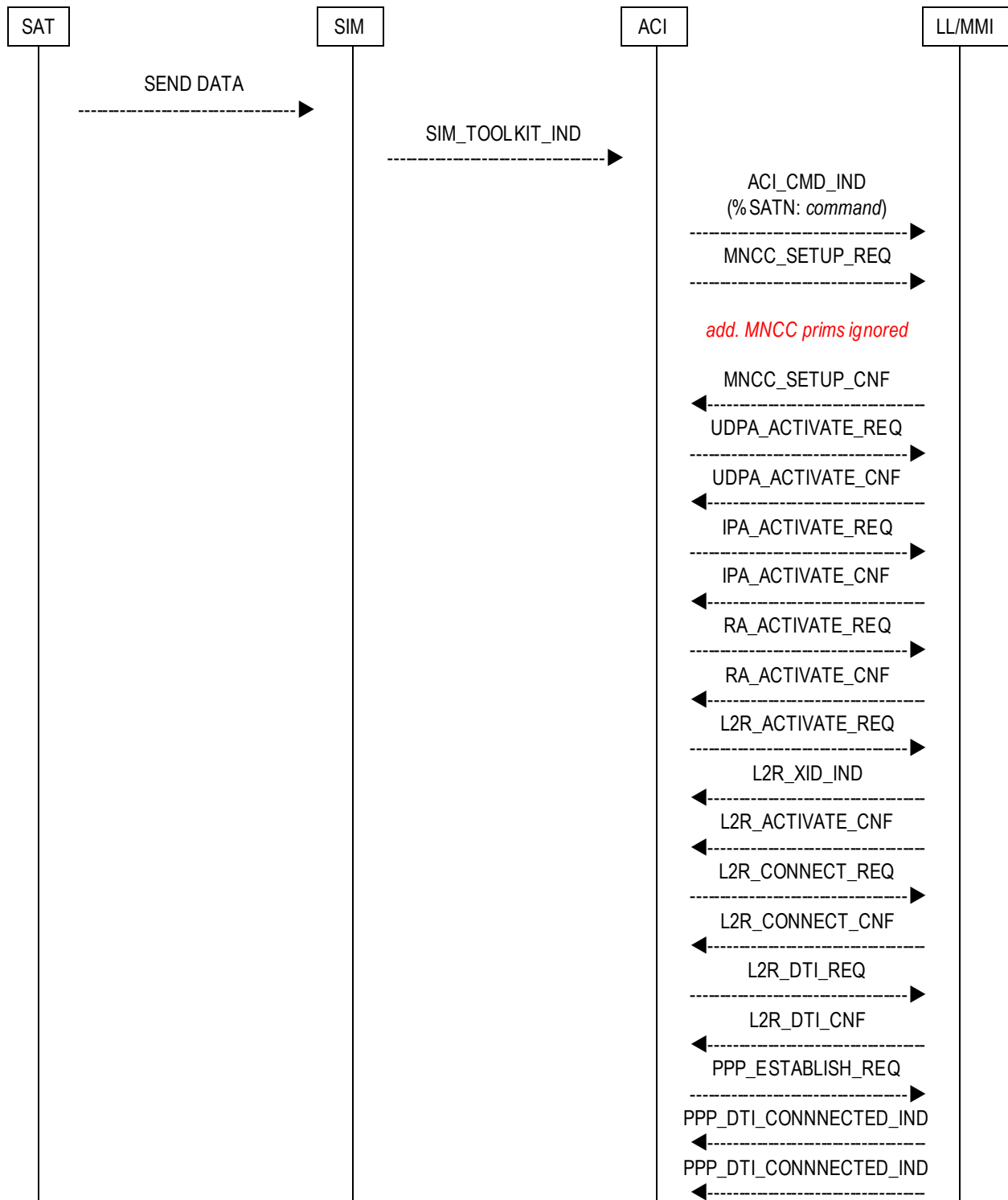
See MSC (11.2.10) as a predecessor for this scenario. After SAT has requested an on demand channel establishment, the first SEND DATA command with the command details flag set to "immediate" will trigger the channel establishment. In this scenario the channel establishment will be rejected by the network. ACI has to close the SIM channel, which is still open. Together with the request to close the channel, ACI passes the cause for the failure "Network currently unable to process command". SIM is in charge to terminate the SEND DATA command properly.

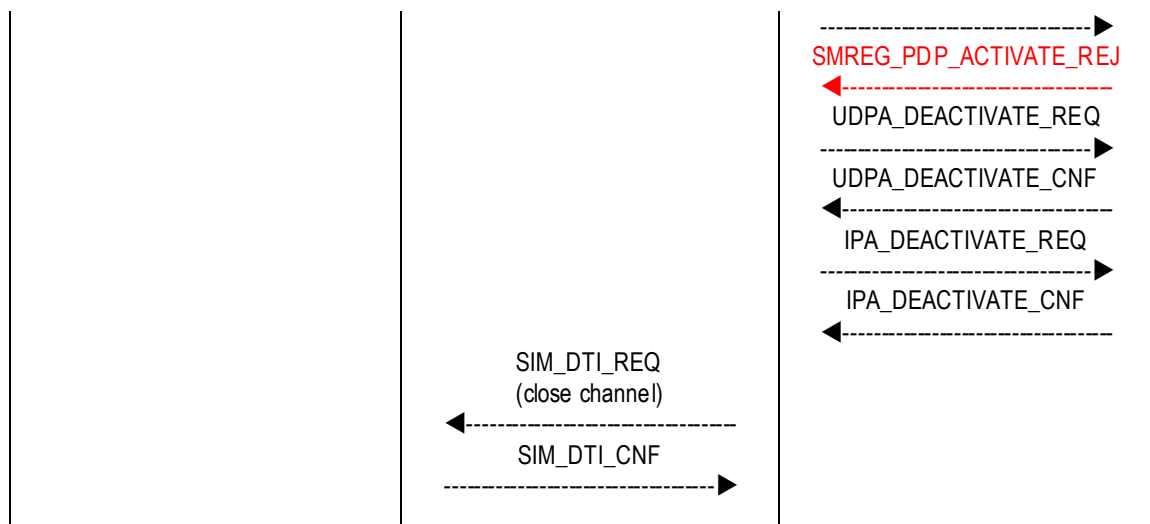


11.3.5 SEND DATA, immediate, CSD, failure within the data chain

See MSC (11.2.10) as a predecessor for this scenario. After SAT has requested an on demand channel establishment, the first SEND DATA command with the command details flag set to "immediate" will trigger the channel establishment. In this scenario the channel establishment will fail due to a problem with the L2R entity. ACI has to release the channel and close the SIM channel, which is still open. Together with the request to close the channel, ACI passes the cause for the failure "ME currently unable to process command". SIM is in charge to terminate the SEND DATA command properly.

In this scenario the L2R entity is the source of the problem. For other data entities the scenario will be similar with minimal modifications. Therefore not all possible combinations are listed here.

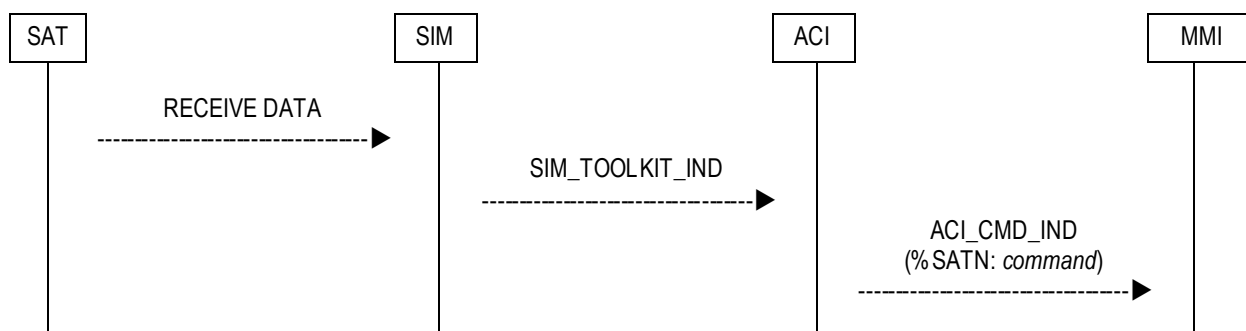




11.4 RECEIVE DATA

11.4.1 RECEIVE DATA

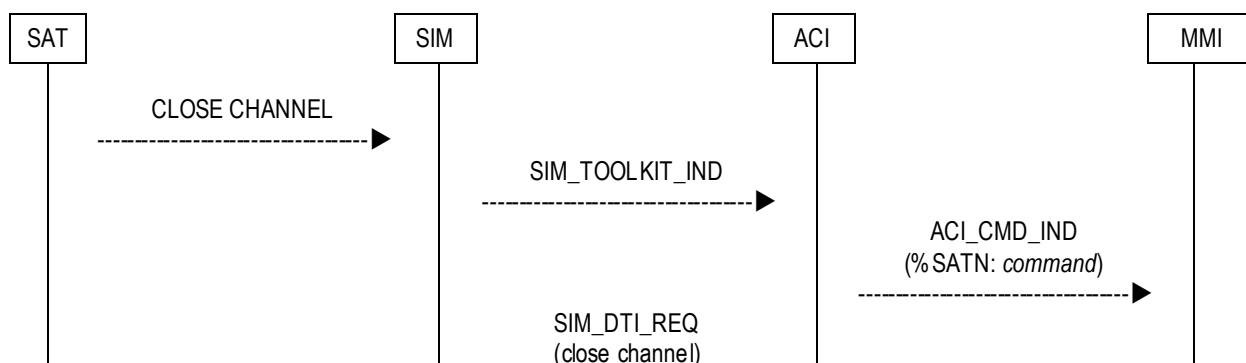
See MSC (11.2.1) as a predecessor for this scenario. A SAT data channel is up and running. In this case all RECIEVE DATA commands will be passed to MMI, without any action by ACI. The handling of the data transmission will be done by SIM. In case of display information within the RECIEVE DATA command, MMI has to act accordingly.

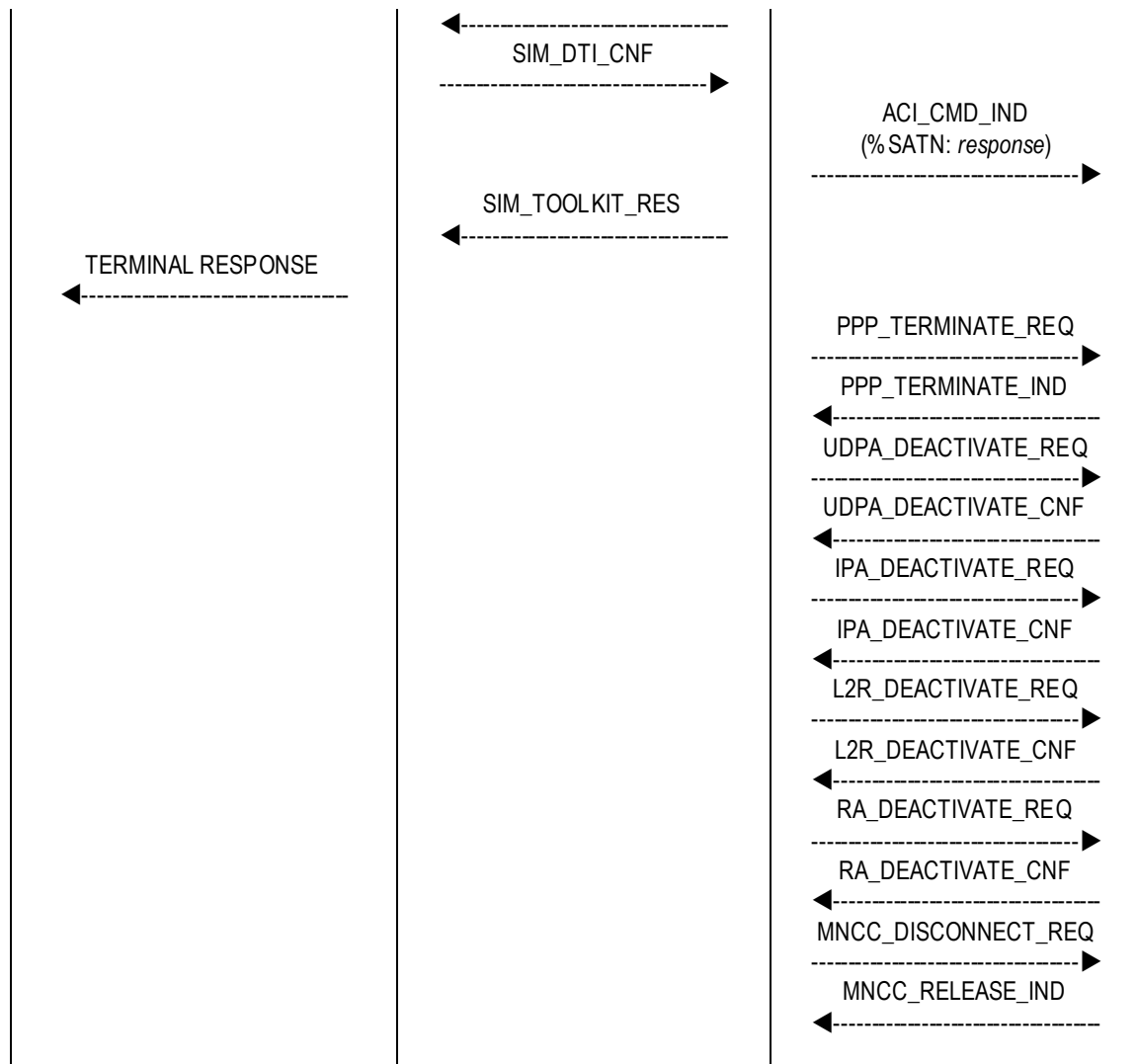


11.5 CLOSE CHANNEL

11.5.1 CLOSE CHANNEL, CSD, UDP

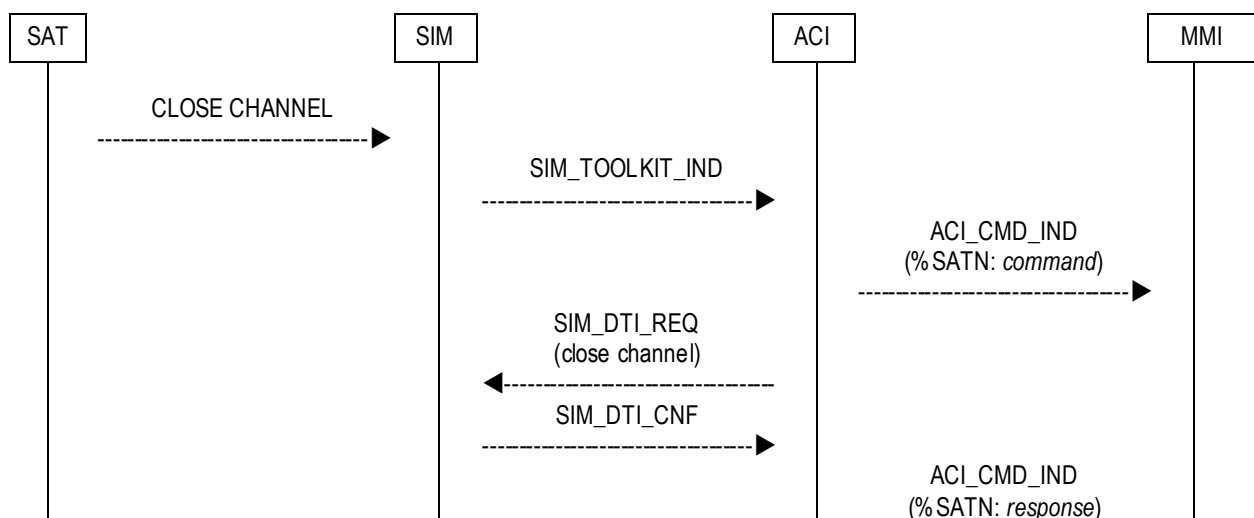
See MSC (11.2.1) as a predecessor for this scenario. A SAT data channel is up and running. SAT wants to close the channel and sends a CLOSE CHANNEL command to ACI. ACI starts to release the channel immediately. After SIM channel is closed the SAT command will be terminated with the cause "Command performed successfully"

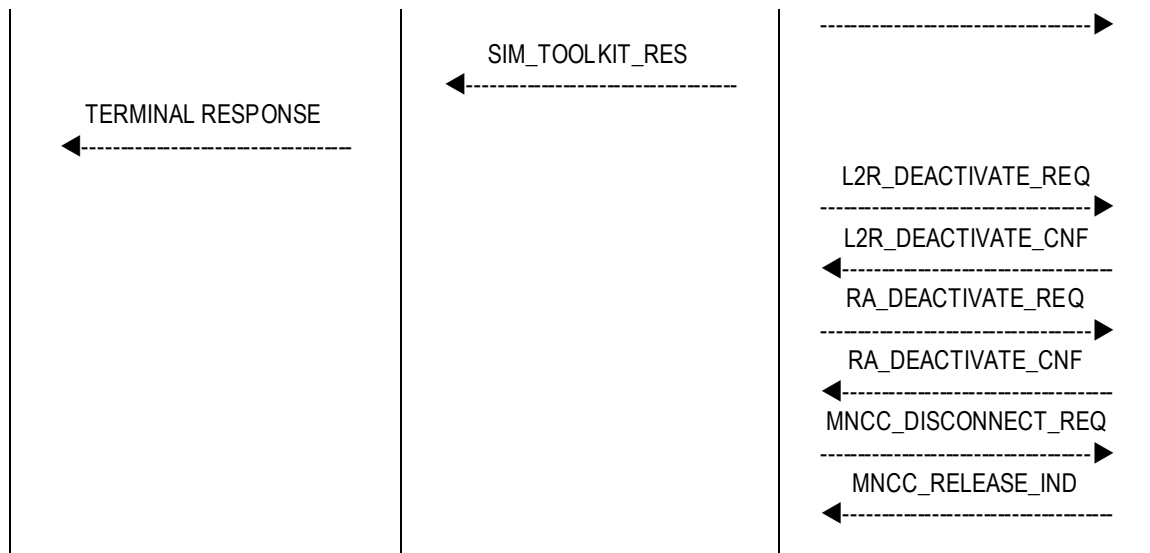




11.5.2 CLOSE CHANNEL, CSD, non-transparent

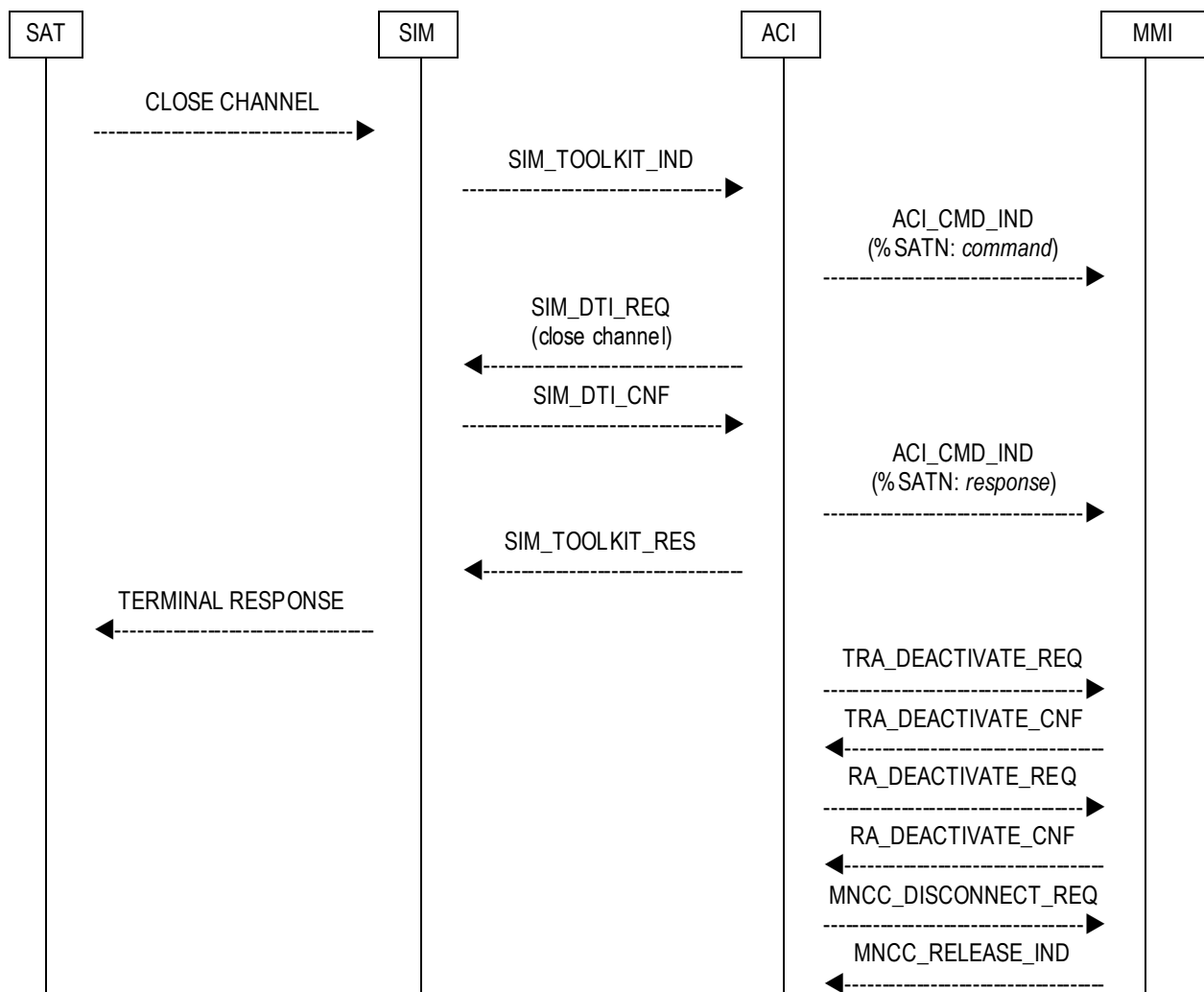
See MSC (11.2.2) as a predecessor for this scenario. A SAT data channel is up and running. SAT wants to close the channel and sends a CLOSE CHANNEL command to ACI. ACI starts to release the channel immediately. After SIM channel is closed the SAT command will be terminated with the cause "Command performed successfully"





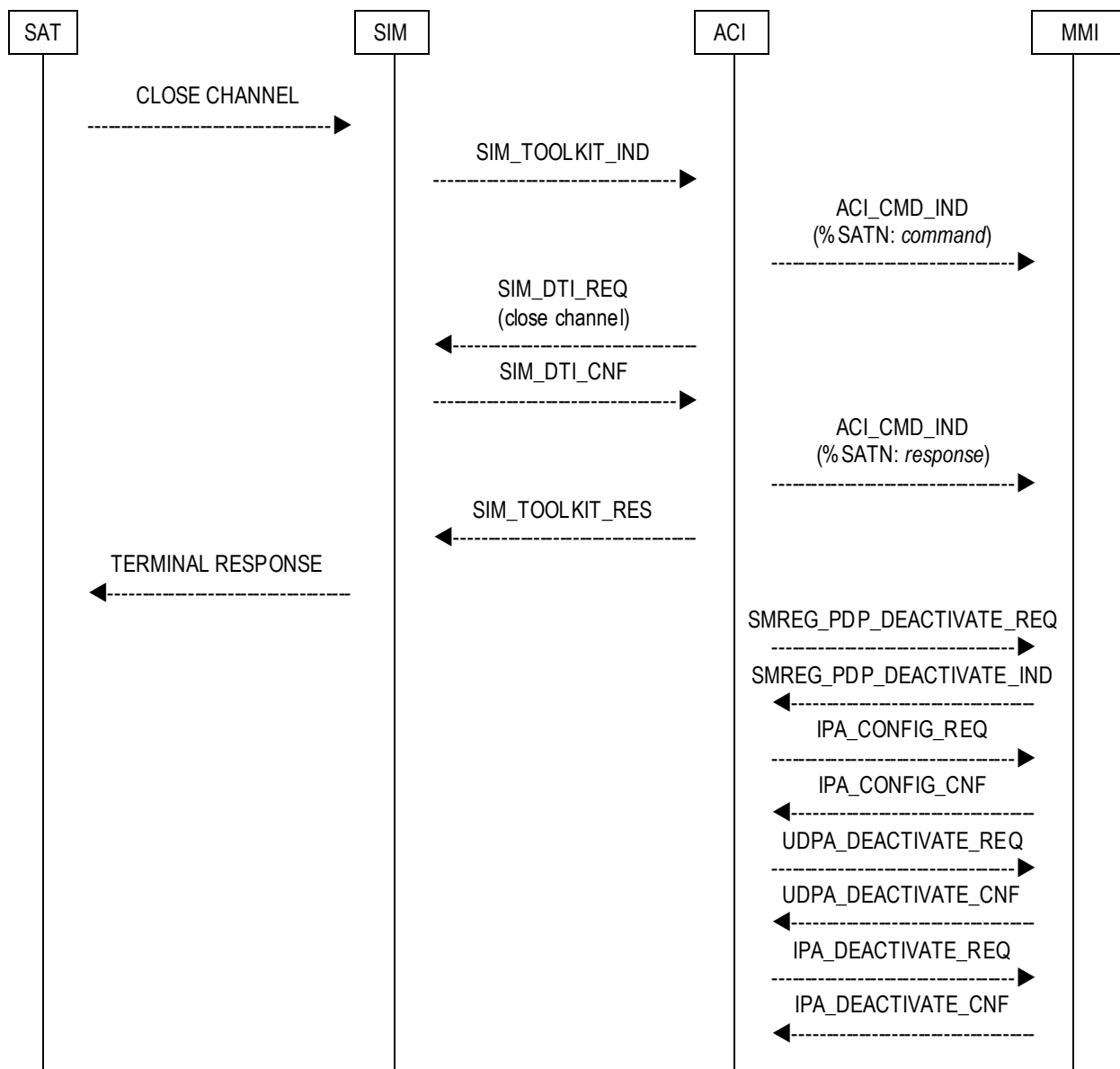
11.5.3 CLOSE CHANNEL, CSD, transparent

See MSC (11.2.3) as a predecessor for this scenario. A SAT data channel is up and running. SAT wants to close the channel and sends a CLOSE CHANNEL command to ACI. ACI starts to release the channel immediately. After SIM channel is closed the SAT command will be terminated with the cause "Command performed successfully"



11.5.4 CLOSE CHANNEL, GPRS, UDP

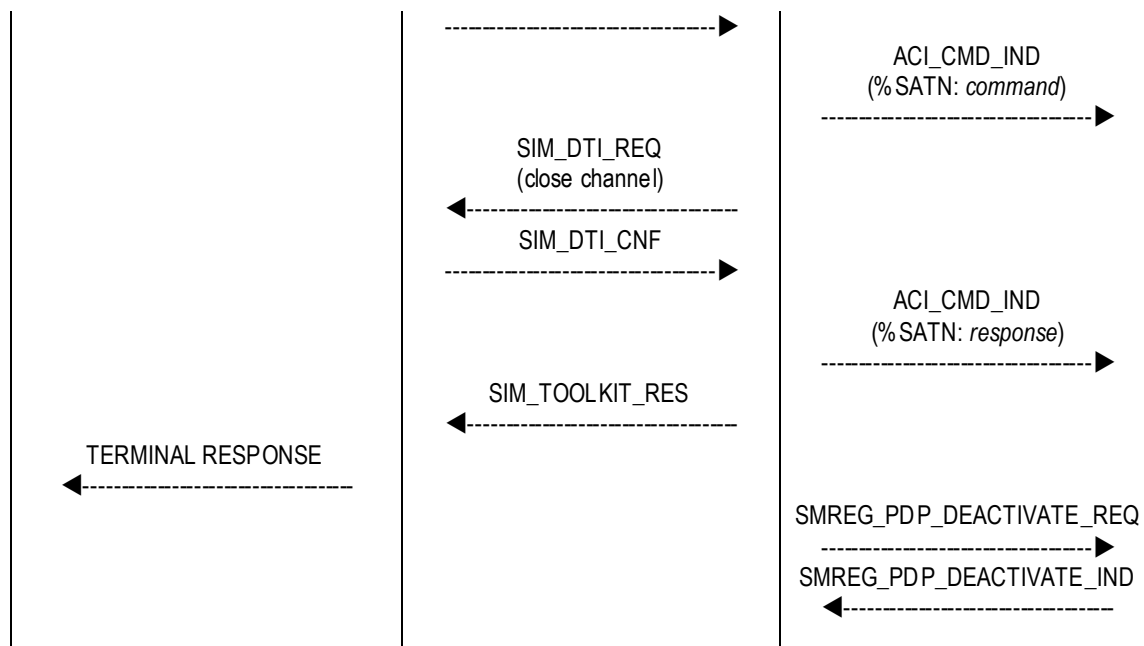
See MSC (11.2.6) as a predecessor for this scenario. A SAT data channel is up and running. SAT wants to close the channel and sends a CLOSE CHANNEL command to ACI. ACI starts to release the channel immediately. After SIM channel is closed the SAT command will be terminated with the cause "Command performed successfully"



11.5.5 CLOSE CHANNEL, GPRS, SMDCP

See MSC (11.2.7) as a predecessor for this scenario. A SAT data channel is up and running. SAT wants to close the channel and sends a CLOSE CHANNEL command to ACI. ACI starts to release the channel immediately. After SIM channel is closed the SAT command will be terminated with the cause "Command performed successfully"

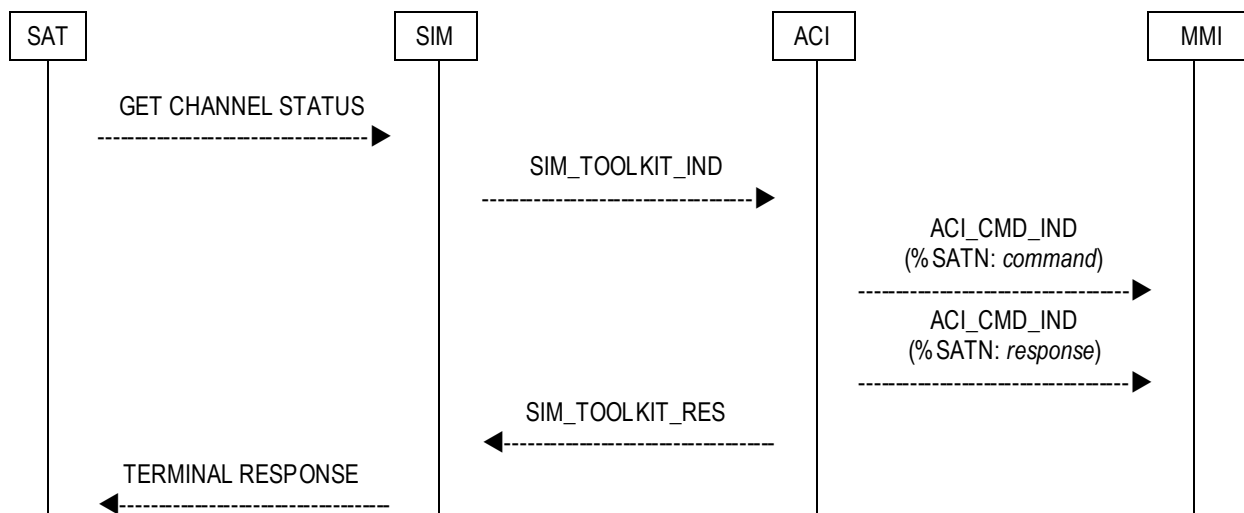




11.6 GET CHANNEL STATUS

11.6.1 GET CHANNEL STATUS

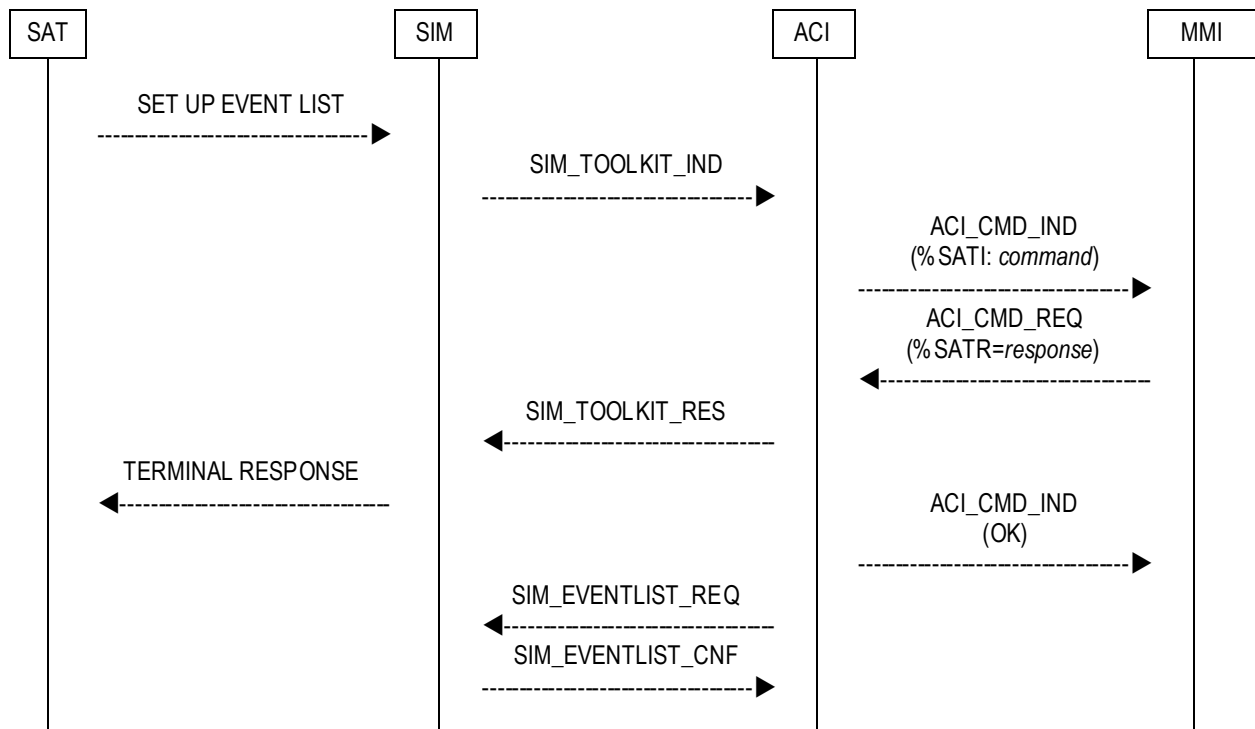
See MSC (11.2.6) as an example predecessor for this scenario. A SAT data channel is up and running. SAT wants to check the channel status and sends a GET CHANNEL STATUS command to ACI. ACI provides the current status and terminates the command with the cause "Command performed successfully"



11.7 CHANNEL STATUS/DATA AVAILABLE EVENT

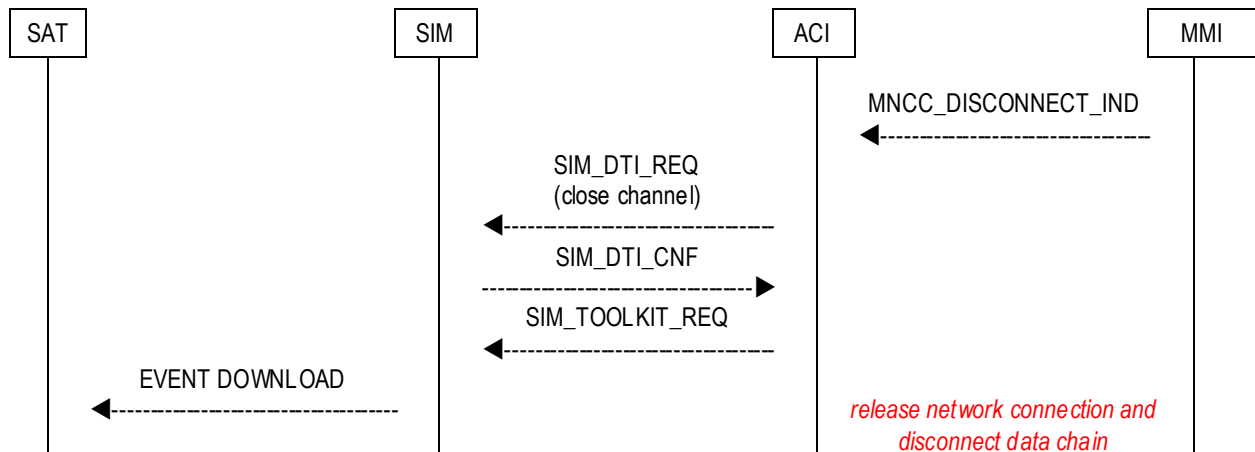
11.7.1 Event Download

SAT indicates the request for the CHANNEL STATUS EVENT and/or the DATA AVAILABLE EVENT with the SET UP EVENT LIST command. Basically this will be handled in the same way as other events are handled. The event list is forwarded to MMI for verification. MMI will then terminate the command with the appropriate result. If the whole event list has been accepted by MMI, the ACI events will be activated too. In case that the DATA AVAILABLE EVENT is part of the list, ACI will pass the status for that event to SIM. The DATA AVAILABLE EVENT will be fully handled by SIM. ACI is only responsible to enable or disable the event in SIM.



11.7.2 CHANNEL STATUS EVENT

The CHANNEL STATUS EVENT will be handled by ACI. If the CHANNEL STATUS EVENT is set by SAT and an already established SAT data channel will break down, ACI indicates this condition by sending the CHANNEL STATUS EVENT to SAT. The following scenario assumes that a CSD channel is up and running. The channel will be terminated by the called party.



Appendices

A. Acronyms

DS-WCDMA Direct Sequence/Spread Wideband Code Division Multiple Access

B. Glossary

International Mobile Telecommunication 2000 (IMT-2000/ITU-2000) Formerly referred to as FPLMTS (Future Public Land-Mobile Telephone System), this is the ITU's specification/family of standards for 3G. This initiative provides a global infrastructure through both satellite and terrestrial systems, for fixed and mobile phone users. The family of standards is a framework comprising a mix/blend of systems providing global roaming. <URL: <http://www.imt-2000.org/>>