High Level Design Description

SECURE ME PERSONALIZATION - ARCHITECTURE OVERVIEW

Document Number:	8462.747.05.001
Version:	1.0
Status:	Approved
Approval Authority:	
Creation Date:	2005-Jul-22
Last changed:	2015-Mar-08 by Oleksiy Kulish
File Name:	hld_sec_sml_architecture.doc

Important Notice

Change History

Date	Changed by	Approved by	Version	Status	Notes
2005-Jul-22	Oleksiy Kulish		0.1	Draft	1
2005-Oct-17	Oleksiy Kulish		0.2	Draft	
2015-Mar-08	Oleksiy Kulish	Thomas Luettig	1	Approved	

Notes:

1. Initial version

Table of Contents

List of Figures and Tables

List of References

1. ETSI TS 122 022 V3.1.0 (2000-07) Personalization of Mobile Equipment (ME); Mobile functionality specification (3G TS 22.022 V3.1.0 Release 1999)

2. ETSI TS 127 007 V3.9.0 (2001-06) AT command set for User Equipment (UE) (3G TS 27.007 version 3.9.0 Release 1999)

3. ETSI TS 151 011 V4.11.0 (2004-03) Specification of the Subscriber Identity Module -Mobile Equipment (SIM-ME) interface (3GPP TS 51.011 version 4.11.0 Release 4)

4. 8462.600.02 - ACI – ME PERSONALIZATION INTERFACE DESCRIPTION

1 Introduction

As per end of 2004 TI protocol stack (e.g. TCS2.1 2.1.1.8) contains a very basic personalization module (SIM Lock module), which covers only requirements, described by [1]. It was mainly caused by the fact that many ME manufacturers have very specific HW, SW, security requirements. Thus TI was supplying only default SIM Lock implementation, which should serve as an example for implementation of customized SIM Lock.

To avoid customer problems in future and make TI Stack more attractive it was decided to implement powerful SIM Lock module which covers most use cases, provides high level of security against intrusion and can be completely used by customers without or with minor changes

Proposed implementation is based on requirements, collected from [1] and different customer requests, coming from previous issues. Some of requirements are shown in Chapter 2.

2 Functional requirements

Functional requirements are shown in Table 2.1

#	Requirement	Comments
TR-0	**General Requirements**	
	ME shall be fully compatible to all mandatory requirements of 22.022 (v.5.0.0. R. 5)	
TR-1	**Personalization Categories**	
TR1.1	NW, NS, SP, CP, SIM Personalization shall be supported	
TR-1.2	Auto personalization shall be supported	
TR-1.3	Extension to Network Subset personalization shall be supported:	
	- subset code can be defined by two other consecutive digits from IMSI	e.g. it is configured that instead of 6th & 7th, the 7th & 8th shall be checked
	- instead of one network subset code, it is possible to define an interval of authorized network subset codes	
	- the 8th digit of the IMSI can be associated with a network subset code or a network subset code interval (defined by 6th and 7th)	e.g. if it is specified that 6th& 7th are in range (20..30) for 8th = 5, imsi 11111255xxxx is accepted, 11111111xxxx is not
TR-1.4	Extension to Service Provider personalization shall be supported:	
	- SP code can be defined by the first two, three or four bytes of the GID1 file	
	- interval of authorized SP code shall be supported	
TR-1.5	Extension to Corporate Personalization shall be supported	
	- CP code can be defined by the first two, three or four bytes of the GID2 file	
	- interval of authorized CP code shall be supported	
TR-1.6	The ME may optionally be personalized to multiple networks, network subsets, SP's, Corporate, SIMs or any combinations thereof	it means, number of IMSIs or other personalization code shall be >1 (15 in current implementation)
TR-2	**Auto Personalization**	
TR-2.1	Auto Personalization shall be supported	

TR-2.2	Test SIMs shall not be treated as a candidate for Auto Lock	It shall never be possible to personalize ME to test SIM
TR-2.3	All categories defined in TR-1.1 shall be available for auto Personalization	
TR-2.4	Auto Personalization shall be configurable	e.g. Lock ME on NW and SP
TR-2.5	Configuration of Auto Personalization shall be a part of MEPD and shall not be changed by the user	Configuration shall be done by config tool only and stored in MEPD. No functional interface for changing of configuration shall be provided
TR-3	**Personalization Verification**	
TR-3.1	Personalization shall be checked for every supported category every time ME is powered on or new SIM is inserted	
TR-3.2	If personalization check fails even for one category ME goes to limited mode (Emergency Calls). Otherwise to full mode	
TR-3.4	If personalization fails user shall automatically be asked to enter keys for each category	That shall be done within MMI. API will be provided for personalization status requesting
TR-3.5	CPIN? Shall give currently used lock (first found) and request entering required control key. Entering correct key shall unlock ME for one session (till next restart)	
TR-3.4	If SIM Personalization category is verified, IMSI is only accepted if NW and NS Personalization passes	e.g. if NW and/or NS locks are set, they must be checked first, then if check passes, SIM is checked
TR-4	**Test SIM**	
TR-4.1	Test SIMs shall be treated - always accepted, always rejected and accepted until once normal SIM is inserted. ME must be configurable for all three Test SIM behaviour models - always accepted - no check is done - always rejected - if any lock category is active, Test SIM will be rejected - accepted until normal SIM is inserted - as soon as normal SIM is inserted, autopersonalization applied to ME	
TR-4.2	CPHS Type test SIMs (IMSI starts with 00101) acceptance shall be configurable - - Always accepted : there is no SIM-ME-LOCK check. The ME is never blocked on a category. - - Always rejected : a SIM-ME-LOCK check is always performed. The ME goes in normal mode only if all personalized categories are unblocked	Accepted/Rejected
TR-4.3	Configuration of Test SIM behaviour shall be a part of MEPD and shall not be changed by the user	Configuration shall be done by config tool only and stored in MEPD. No functional interface for changing of configuration shall be provided
TR-5	**Personalization Process**	

TR-5.1	Personalization cycle types according to ETSI spec shall be supported (via keypad, SIM info, manufacturer defined etc) - personalization over modem SW shall be provided - personalization over direct access to MEPD with help of config Tool shall be provided - API for personalization over the user interface shall be provided	standard CLCK commands manufacturer personalization
TR-5.2	It shall be possible to personalize ME to all categories by entering a **new** key	ETSI behavior
TR-5.3	It shall be possible to personalize ME to any category, which already contains PD and key by asking user to enter a key. Personalization succeeds if entered and stored keys match. If no key was stored, entered key shall be stored without comparison.	Proprietary behavior
TR-5.4	Configuration of Personalization process behaviour shall be a part of MEPD and shall not be changed by the user	Configuration shall be done by config tool only and stored in MEPD. No functional interface for changing of configuration shall be provided
TR-5.5	API shall provide the possibility to personalize ME only on standard categories. All extended categories are treated as standard	e.g. lock on range, on 7th&8th digits shall be locked/unlocked as NS
TR-5.6	While executing SIM Personalization, new IMSI shall be added to the list of SIM personalization code group	e.g. user sets up anti theft, protection the IMSI of currently inserted SIM is added to MEPD
TR-5.7	It shall not be possible to personalize ME if no SIM card inserted	
TR-6	**De-personalization Process**	
TR-6.1	De-personalization for every single category shall be supported	API for all de-personalization categories shall be provided
TR-6.2	De-personalization for combined category shall be supported - slave categories preserve their own control key and can be independently locked or unlocked by the user - slave categories (SP or C) cannot be independently locked/unlocked by the user	e.g. If ME was personalized to combined categories, unlocking one of them shall automatically mean unlocking another. It is not secure and disagrees TS22.022 - thus unlocking all categories with one PW would mean that PW is same for all categories. But 22.022 specifies that PW shall be unique.
TR-6.3	De-personalization process shall be done via control keys comparing	
TR-6.4	For each failed de-personalization process failure counter shall be increased	
TR-6.5	For each succeeded de-personalization process failure counter shall be reset	

TR-6.6	Re-setting of failure counter shall be done similar to de-personalization process	e.g. By comparing entered and stored personalization key
TR-6.7	It shall be configurable to forbid all unlocking/changing pass-word attempts by user/modem SW, as if entered control keys were wrong	LAM requirement - if the user tries to unlock and enters the key, even if key is correct, action is ignored and "wrong key" message returned
TR-6.8	It shall not be possible to de-personalize ME if no SIM card inserted	
TR-7	**Data Storage**	
TR-7.1	MEPD format shall be specified by TI during design phase	
TR-7.2	Size of MEPD shall be limited to 8Kbytes	
TR-7.3	Secure part of MEPD shall be stored encrypted and shall be protected from unauthorized access	Shall be dependent on Modem Security Driver implementation
TR-8	**Personalization Keys**	
TR-8.1	Specific locking keys could be used for locking unlocking. Be-haviour shall be configurable	lock key shall be different from unlock. Up to TI, which way to use, separate stor-ing or generating from un-lock key. Consider Config tool dependency
TR-8.2	Keys generating	covered by separate doc
TR-8.3	Key truncation to 8 byte	e.g. it shall be configurable which length shall be checked - standard or 8 bytes
TR-8.4	Standard Key length shall be 6-16 bytes for PCK and 8-16 for other control keys	
TR-9	**Security Requirements**	
TR-9.1	MEPD Data should be secured with confidentiality	Exact implementation is up to customer implementation of Modem Security Driver
TR-9.2	Secure personalization info (e.g control keys) shall be kept separately from non-secure data (e.g. timer value etc) and shall not be accessed by ACI directly	Exact implementation is up to customer implementation of Modem Security Driver
TR-10	**Supplementary Information**	
TR-10.1	Failure Counter concept shall be used to limit number of failing de-personaliz ation attempts - every time de-personalization fails, counter is increased. When counter reaches its maximal value, de-personalization process cannot be run anymore. ME shall be returned to factory for unlock.	
TR-10.2	Max Value of the counter shall be stored together with other MEPD and thus is configurable	

TR-10.3	Handling of MMI behaviour of the counter (e.g. Increasing timer between de-personalization attempts) shall be provided	shall be provided by customer MMI, if required
TR-10.4	Failure counter shall be also configurable for unlimited value	
TR-10.5	It shall be possible to interrogate Supplementary Information	API for retrieving any supplementary info values. Concept of flexible API is developed in LLD

TR-11 SW Requirements

TR-11.1	MMI shall have no direct access to MEPD	API for retrieving only certain information will be provided
TR-11.2	MNC of length 3 shall be supported	
TR-11.3	GIDx at least of length 5 shall be supported	for proprietary locks it must be necessary to check more than 1 byte of GIDx
TR-11.4	Number of attempts to reset Failure Counter shall be configurable only by config tool	No API will be provided
TR-11.5	Following functionality will be required for ACI-MFW communication - +CLCK: for lock/unlock/interrogate standard categories - %XXXX: for lock/unlock/interrogate proprietary categories - %XXXX: for get supplementary info - +CPWD: for password change for standard categories - %XXXX: for password change for proprietary categories - +CPIN: for passing required Control key for unlock for one session During design phase interface can be extended/changed	
TR-11.6	Maximal number of personalization code groups (IMSI, MNC/MCC etc) for each category shall be limited to 20 for NS, 5 SP and 1 CP	
TR-11.7	If MEPD contains already maximal number of personalization codes and user wants to personalize ME to new SIM card with new code, the oldest (first added) code will be replaced with newest one	FIFO way

TR-12 Memory Requirements

TR-12.1	For memory optimization the max size of MEPD can be decreased during design/implementation phase	

TR-13 Documentation Requirements

TR-13.1	LLD or comparable documents for following modules shall be	

	provided: ACI, MSD, MFW, MMI, Config Tool	
TR-14	**User Interface Requirements**	
TR-14.1	UI for SIM personalization shall be provided	
TR-14.2	UI for entering a control keys for each locked category to un-block ME for one session	e.g. as a response for CPIN
	MMI shall provide following features: - UI for ME locking to NW, NS, SP, CP, SIM - UI for ME unlocking NW, NS, SP, CP, SIM - UI for status check of NW, NS, SP, CP, SIM locks - UI for password changing of NW, NS, SP, CP, SIM locks	
TR-14.6	- UI for failure counter reset	
TR-15	**Config Tools**	
TR-15.1	Config tool shall be able to create MEPD and save it into local File	
TR-15.2	Output MEPD format of Config tool shall be synchronous with MEPD format used internally by low-level entities	

3 Architecture

3.1 Current architecture

As it was already mentioned, that current SIM Lock implementation does not cover majority of use cases, has serious security problem and serves actually as an example for customized SIM Lock implementation.

Default SIM Lock module architecture is shown on Fig. 3.1. For detailed description please refer [4].

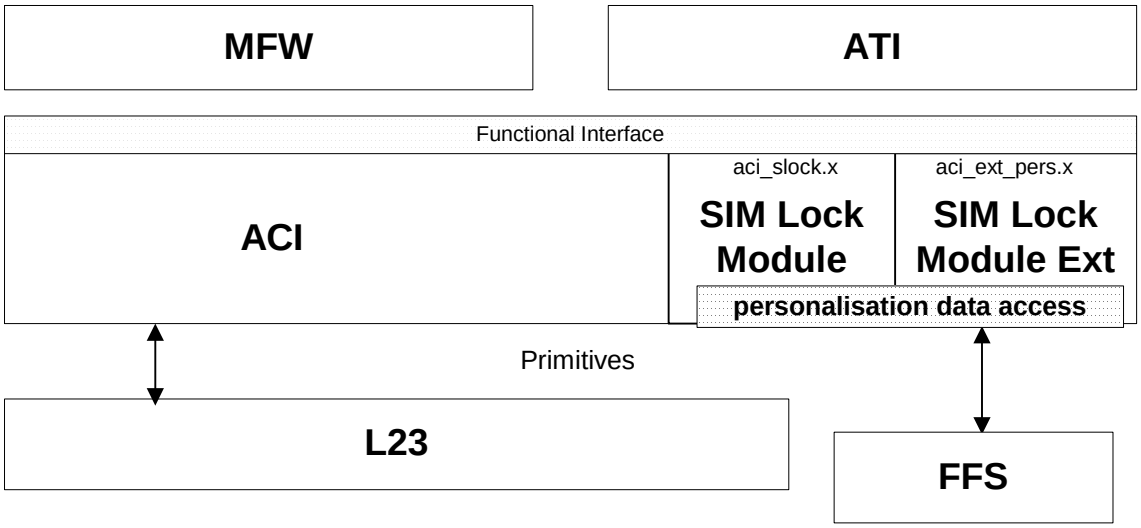

Fig. 3.1 Current ME Personalisation Architecture

3.2 Enhanced Architecture

It is proposed to enhance SIM Lock functionality by re-designing SIM Lock module.

Enhanced SIM Lock architecture is shown on Fig. 3.2

Following constraints shall be applied to the architecture:

1. All MMI applications shall run in MMI space
2. Communication of MMI to ACI shall be done through command handlers (e.g. no direct SIM Lock module calls shall be done from MMI)
3. SIM Lock Module shall consists of SIM Lock Module itself (which provides generalized part) and SIM Lock Module Extension (which contains the requirements specific logic and can be re-implemented by customers)
4. Both SIM Lock and SIM Lock Ext module can communicate to other entities (e.g. SIM)
5. It is desirable that only SIM Lock Ext (not a SIM Lock) communicates to Secure Driver
6. Exceptionally SIM Lock and SIM Lock Ext might be accessed directly from ATI Extension for implementing some specific AT commands to avoid ACI changes.
7. Secure driver is responsible for delivering Personalization Data to SIM Lock Module, updating its status according to SIM Lock request, doing all Security procedures such as MEPD decoding, Control Keys comparison etc. For more details please see 3.3

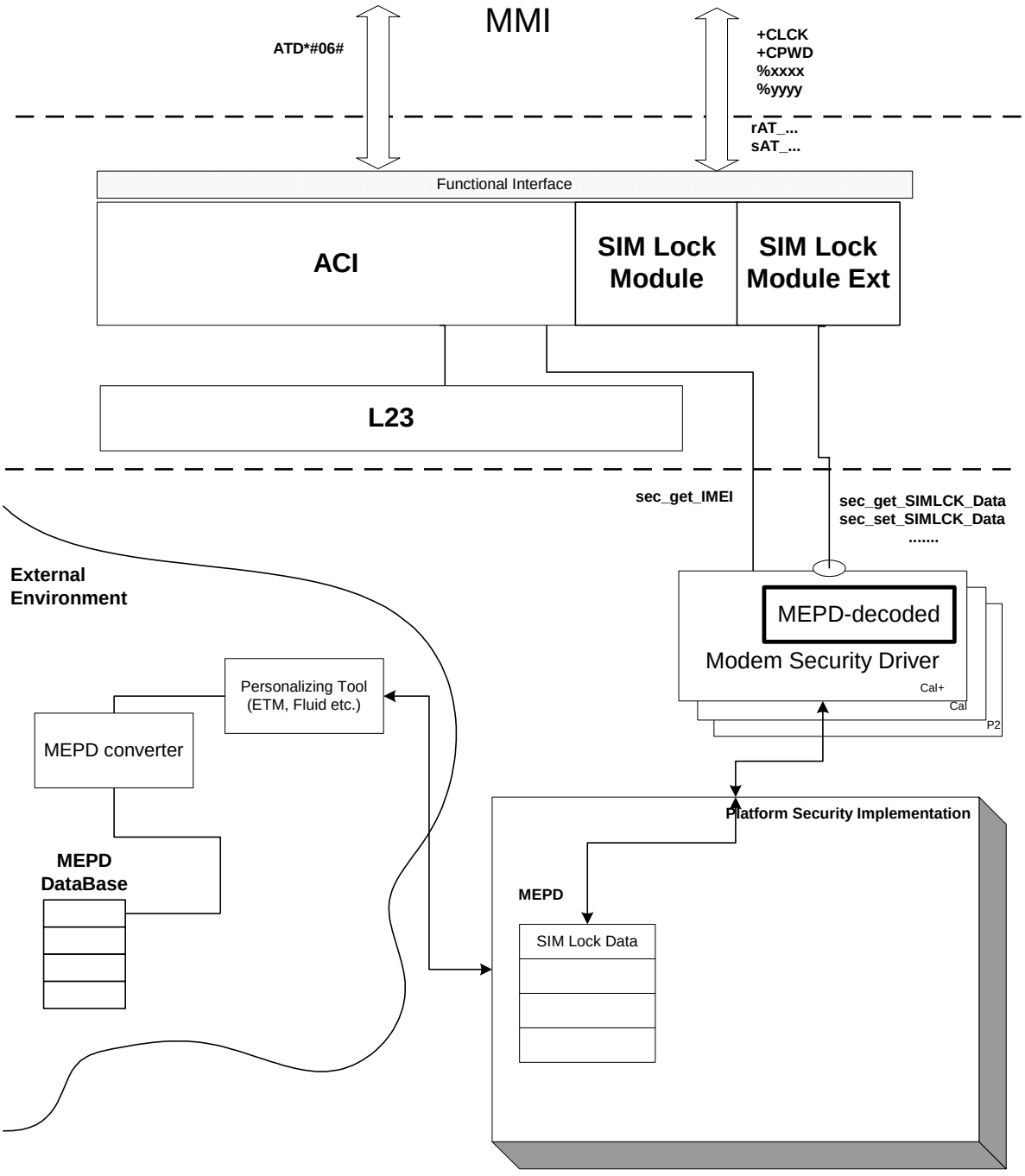

Fig. 3.2 Enhanced SIM Lock Module Simplified Architecture

8. Personalizing Tool is a mean to save Personalization data into non-volatile memory

Functional Blocks:

1. MEPD Database – Customer will store settings for ME personalization in its own database in its own format.
2. MEPD converter – to convert ME Personalization settings from customer format to MEPD format, used by Modem Security Driver, MEPD converting tool will be needed. It shall be created by the customer after clarifying all requirements to ME Personalization functionality and TI MEPD format.
3. Personalizing Tool – tool used to store MEPD on ME storage (FFS, secure ROM etc)
4. Platform Security Implementation – set of SW and HW means to manage storage, access and altering MEPD on different TI platforms.
5. Modem Security Driver – SW module to provide access from L23 (mostly ACI) to MEPD. MSD is platform and Customer requirements dependent. Thus customer has always a possibility to create his own MSD, realizing required encryption algorithm.
6. SIM Lock Module Ext – extension of ME Personalization module which contains major logic of ME personalization. Extension provides an API towards SIM Lock Module itself. Customer has always possibility to re-implement SIM Lock Ext to fit his requirements, leaving other parts of SW untouched
7. SIM Lock module – mostly this is an adapter between SIM Lock Module Ext and other modules of L23
8. ACI – Access Controller Interface, SW module for communication to upper layers
9. L23 – Other modules of Layer 2,3
10. MMI – Upper layer, realizing interaction to user/applications

3.3 Platform Security Implementation

New implementation shall be used on different platforms, used by TI, with minor changes. To separate L23 from platform dependency, Modem Security Driver shall be introduced. Thus, Modem Security Driver will not only serve as module to encrypt/decrypt MEPD data, stored in ME, but will be responsible for usage of Platform Security Means properly.

Some features of different platforms are shown in Table 3.3

Table 3.3

Platform	**Specifics**
Calypso+, Locosto	secure ROM available
Calypso	security is provided by usage of key, “generated” by DSP
Sedna	two processors are connected by HW interface, thus one of the tasks of Secure Driver could be to copy MEPD from ARM11 to ARM9 into MEPD’;
Neptune	one processor solution
P2	communication is provided via Shared Memory

Platform Security Implementation for different chipsets are shown on Fig. 3.4

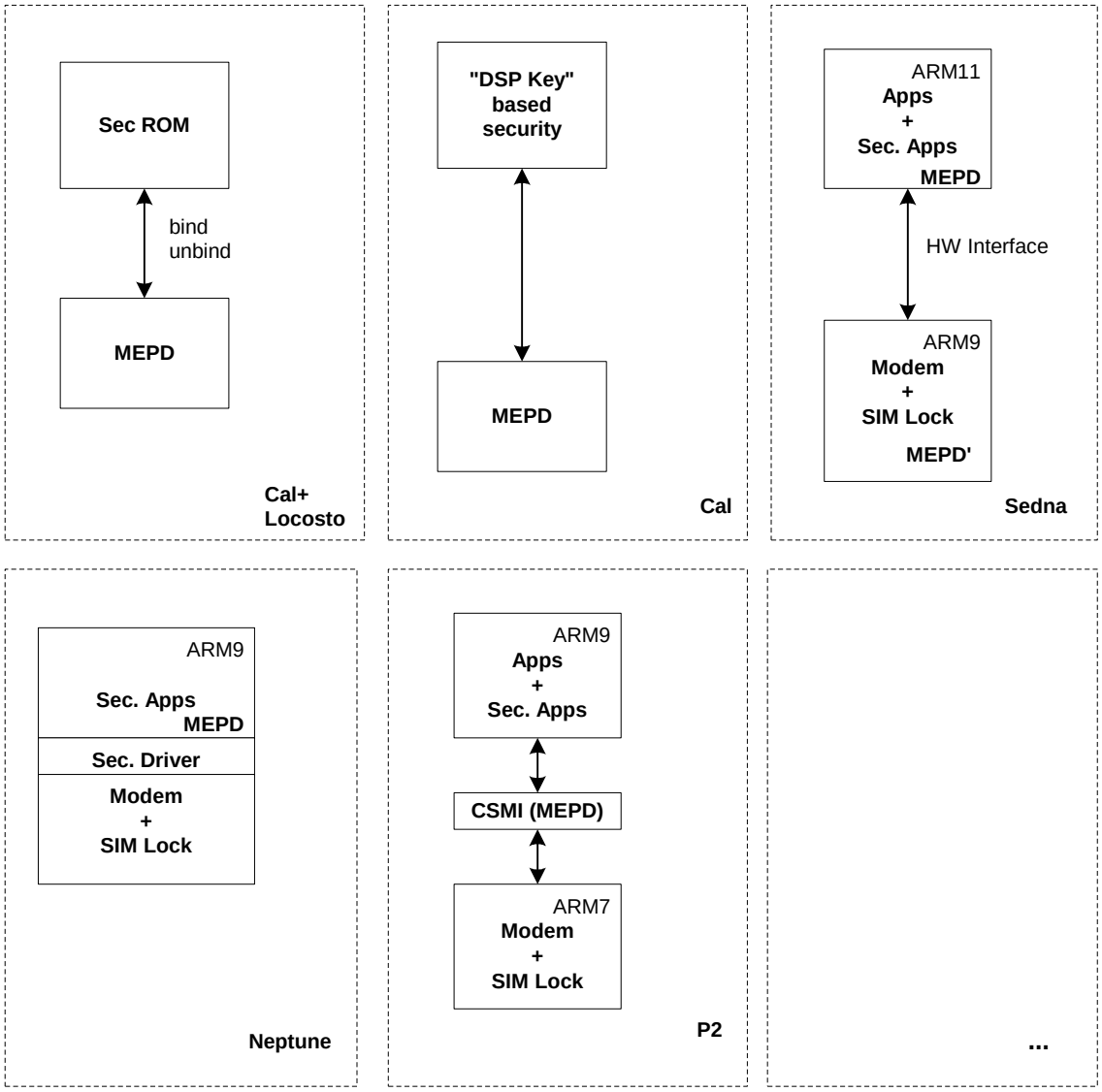

Fig. 3.4 Platform Security Implementation

4 Use Cases

To understand functionality of ME Personalization and make design easier, following Chapter describes Use Cases for main blocks. Please keep in mind that Use Cases may differ from real implementation and could be refined after complete integration of ME Personalization.

4.1 MMI Use cases

Following general MMI Use Cases shall be supported:

1. ME Lock
2. ME Unlock
3. Permanent Unlock (extension of ME Unlock) – currently not considered

4. Status Interrogation
5. Password Changing
6. Failure Counter Reset
7. Powering On
8. Supplementary Info Request/Setup

All other actions/features (lock on range, on GID etc.) shall not be accessible from UI and be available only through MEPD configuration or Flashing Tool access.

MMI Use Case Diagram is shown on Fig. 4.1.1

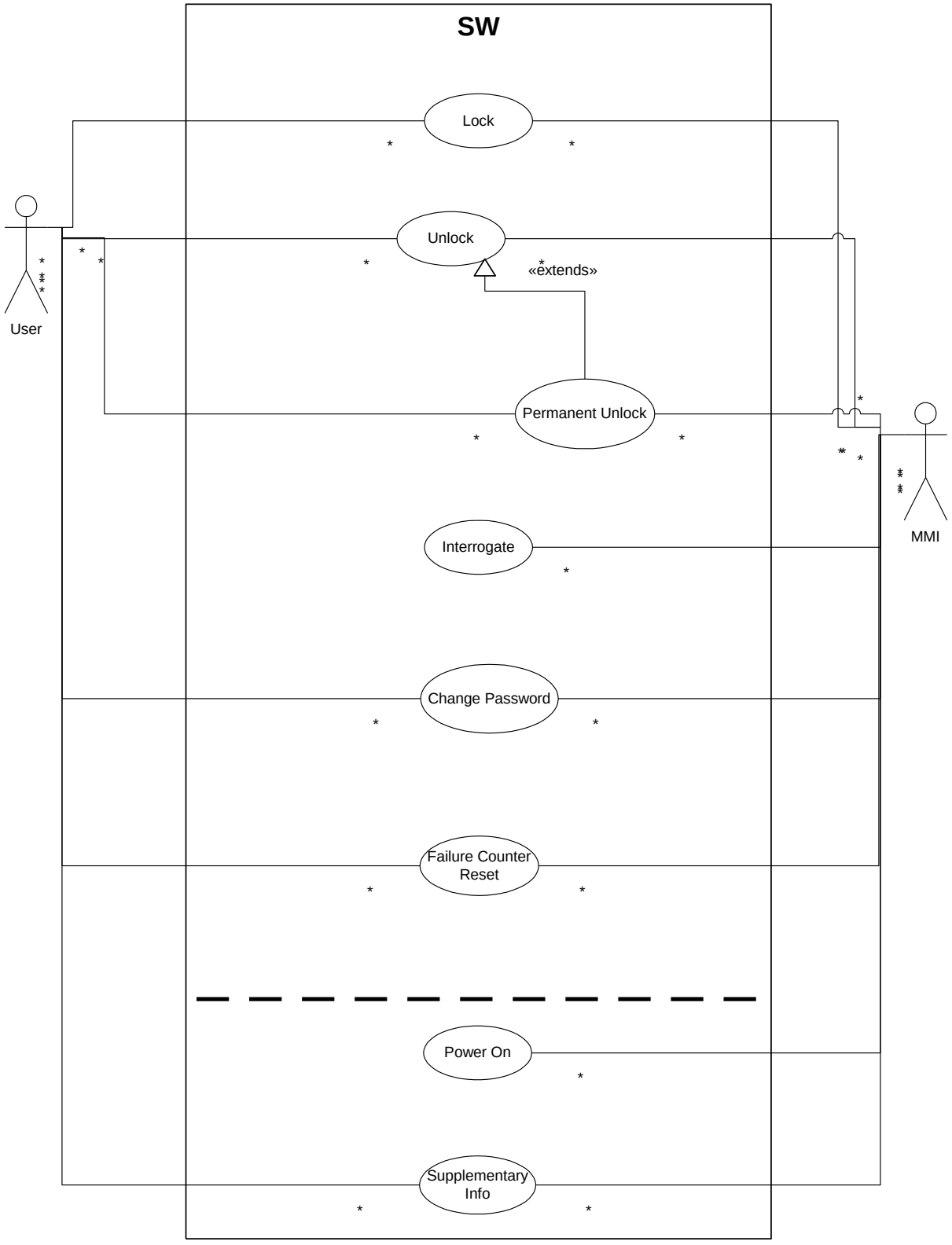

Fig. 4.1.1 MMI Use Case

Fig 4.1.2 gives MMI Use Case Description:

Use Case: **Lock**
Initiator: User
Preconditions: ME is unlocked. SIM card is present
Categories: NW, NS, SP, CP, SIM
Primary Sequence:
1.User selects Menu Item for ME lock of needed category
2. ME asks Control Key for Category
3. If Control Key if correct (see TR-5.2, 5.3)ME gets locked
4. User is informed about operation success
Secondary Sequence:
3. If Key is wrong, user is asked again
4. Error processing if failure happens

Use Case: **Unlock, Permanent Unlock**
Initiator: User
Preconditions: ME is locked.
Categories: NW, NS, SP, CP, SIM, Combined; Magic
Primary Sequence:
1.User selects Menu Item for ME unlock of needed category
2. ME asks Control Key for Category
3. If Control Key if correct ME gets unlocked
4. User is informed about operation success
Secondary Sequence:
3. If Key is wrong, user is informed about failure counter increment and asked for key again
4. Error processing if failure happens

Use Case: **Interrogate**
Initiator: User (indirect)
Preconditions:
Primary Sequence:
1.User enters Menu for ME lock/unlock
2. MMI checks status of locks and enables/disables Menu Items for different Categories

Use Case: **Failure Counter Reset**
Initiator: User
Preconditions:
Primary Sequence:
1.User selects Menu Item for Failure Counter Reset
2. ME asks Control Key
3. If Control Key if correct Failure Counter gets reseted
4. User is informed about operation success
Secondary Sequence:
3. If Key is wrong, user is asked again. Static counter is increased (counter for failure of Failure counterreset)
4. Error processing if failure happens

Use Case: **SIM Lock check on Power On**
Initiator: User (indirect, by switch ME on)
Preconditions: SIM card is inserted
Primary Sequence:
1. User switches ME on
2. ME checkes all possible lock categories
3. If ME is not blocked (not locked or SIM satisfies pass conditions) ME is started as usualy
Secondary Sequence:
3. If ME is blocked user is asked for Control Keys for all locked Categories sequentualy
4. Error processing if failure happens

Use Case: **Supplementary Info**
Initiator: User
Preconditions:
Primary Sequence:
1. User selects Menu Item for Supplementary Info displaying
2. ME shows requested info (e.g. Failure Counter, Lock Status etc.)

Use Case: **Change Password**
Initiator: User
Preconditions: ME is unlocked.
Primary Sequence:
1.User selects Menu Item for password change for needed lock category
2. ME asks Old Control Key for Category
3. If Control Key if correct ME asks to enter and re-enter new Key
4. User is informed about operation success
Secondary Sequence:
3. If Key is wrong, user is informed about failure counter increment and asked for key again
4. Error processing if failure happens

Fig. 4.1.2 MMI Use Case Description

To implement above mentioned use cases in MMI, MFW shall provide access to ACI API. Thus, MFW shall map all ACI APIs into format, accepted by MMI and grant the access to MMI. Syntax of MFW API will be developed during ACI/MFW/BMI design phase.

4.2 ACI Use Case

ACI Use Cases can be interpreted as AT commands actions (API description). Fig. 4.2.1 shows AT Commands based MMI-ACI interaction.

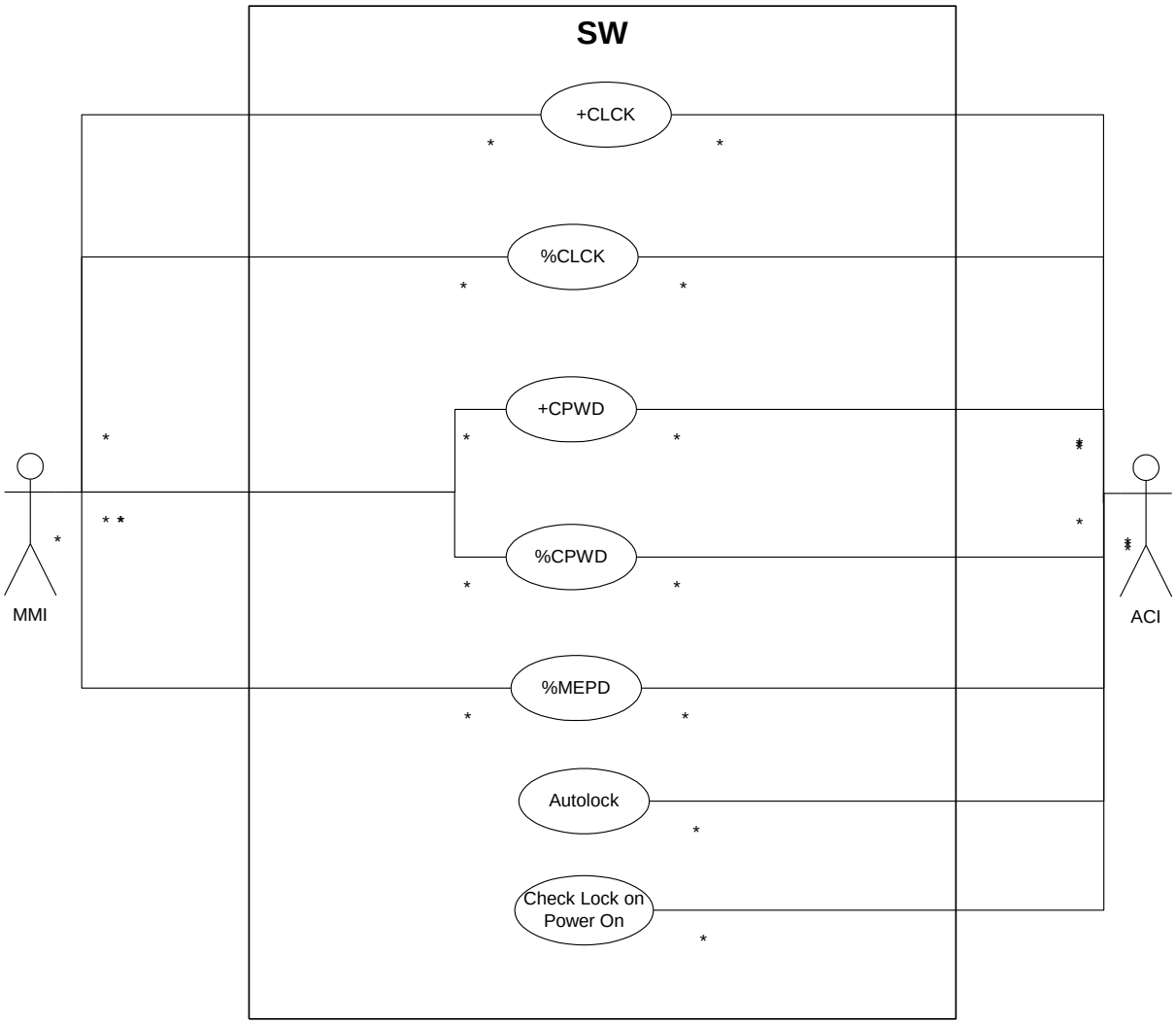

Fig. 4.2.1 ACI Use Case

ACI API description is shown on Fig. 4.2.2
The +CLCK and +CPWD commands are standard AT commands, described in [1]. The commands %CLCK and %CPWD are proprietary TI commands for proprietary locks handling. It is also possible to extend +CLCK and +CPWD instead of introducing new commands – to be finalized in ACI LLD.

+CLCK
Description: Standard AT command, described in 07.07
Preconditions:
Params:
please refer 07.07 for details

%CLCK
Description: proprietory AT command for additional locks
Preconditions:
Params:
fac - facility
"FC" -failure counter reset (unlock)
"MG" - magic unlock
"xx" - TBD: proprietory locks
mode
0 - unlock
1 - lock
2 - interrogate
3 - permanent unlock
4 - lock with new password (TR-5.2)

+CPWD and %CPWD are similar to CLCK
Description:
Preconditions:
Params:

%MEPD
Description: proprietory AT command for requesting/ setting supplementary personalization data.
It is proposed that function operates with (void *) and lenght of delivered data. Data shall be casted in MMI. It would give flexibility and expandability to SIM Lock implementation and prevent ACI changing for SIM Lock Customization

Preconditions:
Params:
void * - pointer to returned data
int* - length of returned data

Example:
```
qAT_PercentMEPD(void **pBuf, unsigned int *len)
{
  struct{int i; char c} t_MEPD;
  t_MEPD.i = 1;
  t_MEPD.c = 'c'
  *pBuf = &t_MEPD;
}
...
main()
{
...
  typedef struct{int i; char c} T_MEPD;
  void * p;
  unsigned int len;
  qAT_PercentMEPD(&p, &len);
  printf("%x%x", ((T_MEPD*)p)->i,  ((T_MEPD*)p)->c);
...
]
```

Of course t_MEPD and T_MEPD shall be identical.

Fig. 4.2.2 ACI API Description

The most challenging part of ACI is internal logic, how the different kind of locks are handled. It will be defined in ACI LLD.

4.3 Modem Security Driver

Although almost all the Personalization logic is kept inside of ACI, because of security reasons there are quite a lot of actions, to be performed in Modem Security Driver. Fig. 4.3.1 shows basic Use Cases for Modem Security Driver. For more details refer Modem Security Driver related documentation.

Corresponding Use Case description is given on Fig. 4.3.2. Thus, Modem Security Driver will need to know

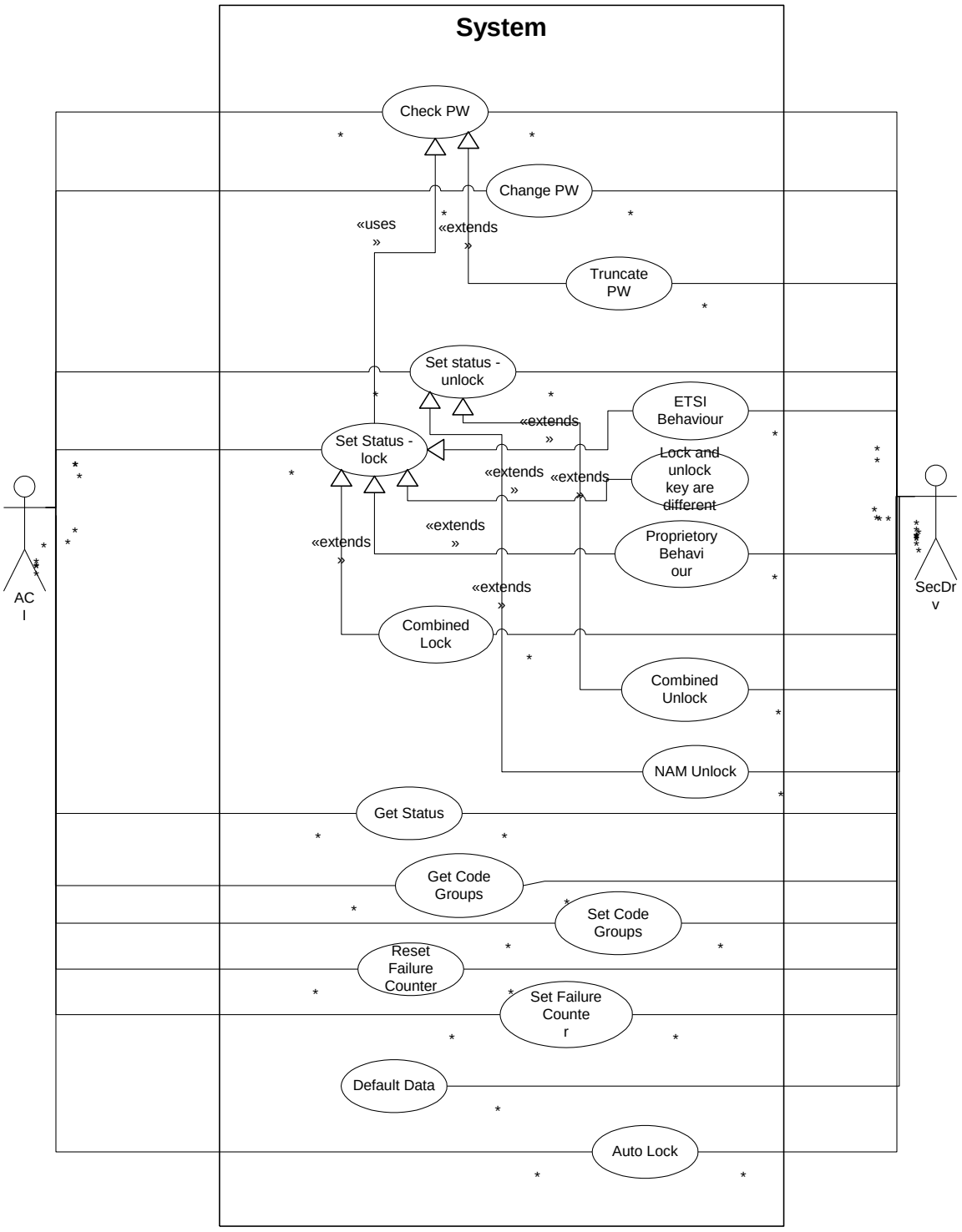

Fig. 4.3.1 Modem
Security Driver Use Case

Use Case: **Change Key**
Initiator: ACI
Preconditions: Category is unlocked .
Categories: NW, NS, SP, CP, SIM, PF, FC
Primary Sequence:
1. SecDrv gets Old Key and New Key
2. SecDrv checks Key constraints (len etc.)
3. Old Key compared with currently saved Key
4. If Old and Saved Keys are identic, Saved Key replaced with New Key
6. SecDrv encrypts and saves new Key
5. Succes returned
Secondary Sequence:
5. If Old and Saved Key differ, error returned
3. If New Key out of constraints, error returned
Config dependency:
1.

Use Case: **Check Key**
Initiator: ACI
Preconditions: .
Categories: NW, NS, SP, CP, SIM, AP, FC
Primary Sequence:
1. SecDrv gets the Key
2. If truncation flag is set, Keys are limited for first 8 bytes
3. Key is checked with a stored one
4. Success returned
Secondary Sequence(Lock and unlock Key differs):
2. If Lock/Unlock Key flag is set...
3. SecDrv converts Key by pre-defined algorithm and checks converted Key with stored one
4. If converted and stored Keys match, success return
Thirdary Sequence :
4. If Keys differ, error returned
Config dependency:
1. Flag - truncation
2. Separate Lock/Unlock key flag

Use Case: **Set status - lock**
Initiator: ACI
Preconditions: Category is unlocked.
Categories: NW, NS, SP, CP, SIM, PF, FC
Primary Sequence (Propr):
1. UC "Check Key" succeeds. If no stored Key present, secondary sequence executes
2. SecDrv changes Status Field of MEPD record
3. Success returned
Secondary Sequence (ETSI):
1. If flag is ETSI, Key is replaced with new
2. Status is changed
3. Success returned
Thirdary Sequence (Combined Lock):
1. Primary sequence succeded
2. SecDrv checkes the "Category Dependency" field of MEPD record
3. Status of dependent categories changed without PW verification (if category data/Key not empty)
Please keep in mind that there are 2 ways of lock - Master/ Slave and Linked. The difference is that M/S can be unlocked independently and Linked cannot. Chiled category can be unlocked only by unlocking Parent category. Thus the status for "linked-child" shall be set not to "locked" but to "linked-locked"
4. If ETSI Flag is set, Keys of combined categories are replaced
5. Success returned
Config dependency:
1. Flag - ETSI or prop behaviour

Use Case: **Default MEPD**
Initiator:
Preconditions: .
Categories:
Primary Sequence:
1. After ME very first start Sec Drv generates default MEPD with only Configuration part of MEPD filled, where configurtion is set to standfard ETSI behaviour

Use Case: **Autolock**
Initiator: ACI
Preconditions: Category Dependency field of Autolock is filled with meaningful data, Status of Autolock set to enable.
Categories:PF
Primary Sequence:
1. On the start of ME wit very firstSIM, the status of all categories within Category Dependency set to active
2. Flag, indicating that it was very first sim is reseted
Secondary Sequence:

Use Case: **Set status - unlock**
Initiator: ACI
Preconditions: Fail Counter less then Max.
Categories: NW, NS, SP, CP, SIM, FC
Primary Sequence:
1. SecDrv gets Key
2. Key is compared with Stored Key
3. If check suceeded, status is changed in MEPD record
4. Success returned
Secondary Sequence (combined unlock):
4. If "Category Dependancy" field of MEPD record contains dependancy date, status of correspondent catogories shall be updated without password check
Thirdary Sequence (NAM Unlock):
1. SecDrv gets Key
2. If NAM unlock impossible Key flag is set...
3. SecDrv returns "wrong password" error
Fourthary Sequence (Combined Lock):
... if Key comparison fails, reffer Fail Counter Increase UC

Config dependency:

Use Case: **Get Status/Get Code Groups**
Initiator: ACI
Preconditions: .
Categories: NW, NS, SP, CP, SIM, PH, FC
Primary Sequence:
1. Status and MEPD records available without Key check just as a Pointer

Use Case: **Fail Counter Increase**
Initiator: ACI
Preconditions: Set Status - Unlock UC is ongoing.
Categories: NW, NS, SP, CP, SIM
Primary Sequence:
1. If Fail Counter equal to Max, SecDrv returns "FailCounterExceed" error
Secondary Sequence:
1. If Keys mismatch, Failure Counter increased, "Wrong Key" error returned

Use Case: **Fail Counter Reset**
Initiator: ACI
Preconditions: .
Categories: FC
Primary Sequence:
1. SecDrv gets FC Key
2. If received Key matches to stored, FC value is reseted to 0 (value is proposed to store in dependency field)
Secondary Sequence:
1. If Keys mismatch, Failure Counter increased, "Wrong Key" error returned

Fig. 4.3.2 Security Driver Use case description

5 MEPD Structure

Proposed MEPD structure is shown on Fig. 5.1

MEPD is spited in three informative areas:

1. MEPD configuration
2. MEPD Records
3. MEPD Keys

And three access areas:

1. Full access
2. Read Only
3. No Access

ACI can see Read Only area “as-is”, but cannot change anything there. MEPD configuration and secure part of MEPD record (status, category dependency) are stored there

Full access area stores the information that is allowed to be freely changed by ACI – code groups.

“No access” area is only accessed by Secure Driver. All control keys are stored there and verified only by Modem Secure Driver. ACI will receive only the result of comparison.

Please refer MSD documentation for more details.

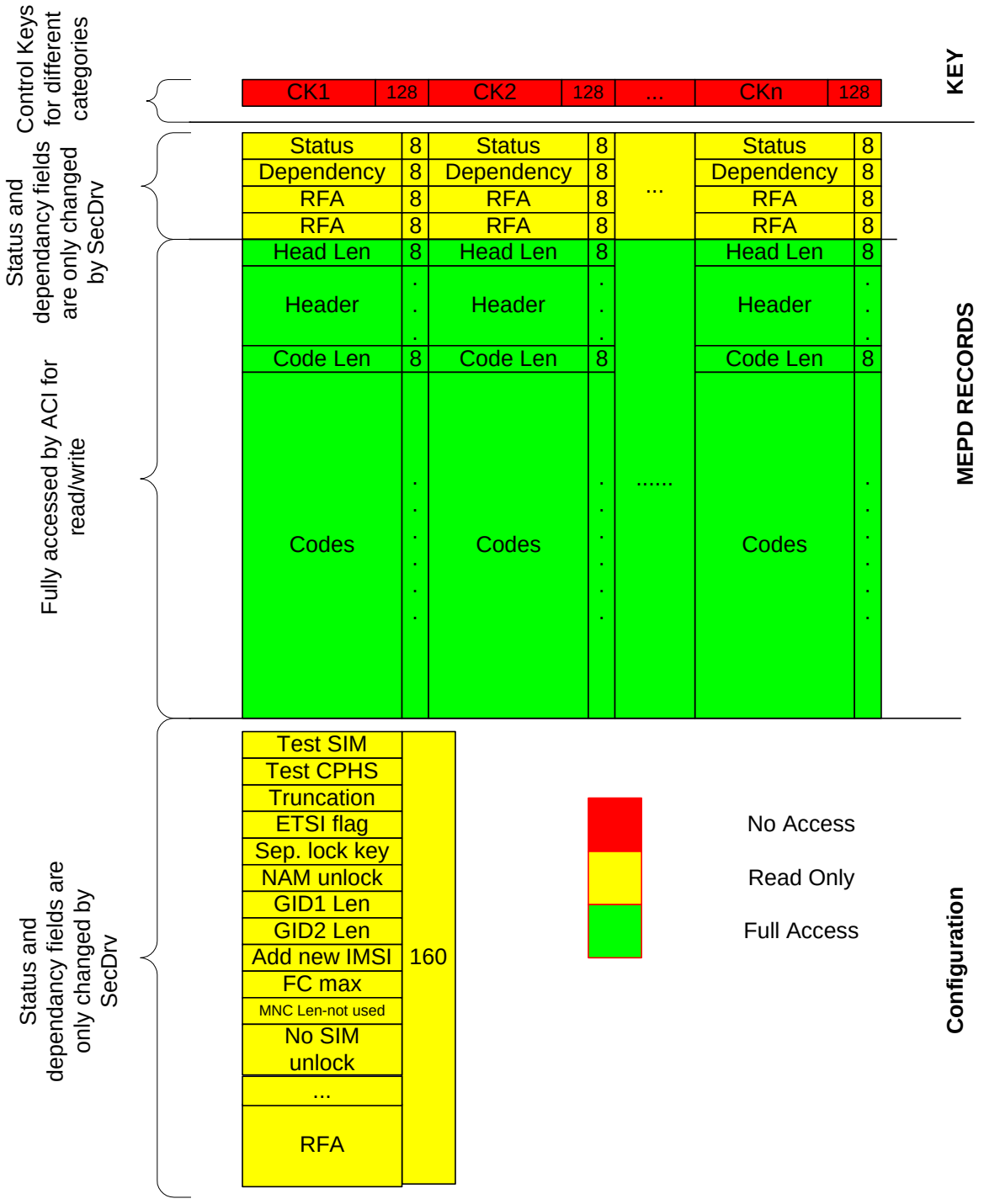

Fig. 5.1 MEPD Structure

Appendices

A. Acronyms

ACI	Access Control Interface
DS-WCDMA	Direct Sequence/Spread Wideband Code Division Multiple Access
DSP	Digital Signal Processor
FFS	Flash File System
IMSI	International Mobile Subscriber Identifier
ME	Mobile Equipment
MEPD	ME Personalization Data
MFW	MMI Framework
MMI	Man Machine Interface
MSD	Modem Security Driver
OTA	Over the Air
RAM	Random Access Memory
ROM	Read Only Memory
SIM	Subscriber Identity Module

B. Glossary

SIM Lock	Module/Functionality providing ME Personalization features, described in [3]
Default SIM Lock	SIM Lock implementation, used in G23 SW as per end of 2003
Enhanced SIM Lock	Newly developed SIM Lock functionality
Customer	Consumer of TI Protocol Stack and ME Personalization Functionality (Phone developer, Operator, Service Centre etc)
User	End User of ME