TELIUM SDK

INTERFACE BETWEEN A VENDING APPLICATION AND A VENDING DLL PROTOCOL

Reference: SMO/SPE-0152
Revision: H
Enter Date: 30/05/2010

Revision Approval: Revision

	Name	Function
Written by:	J. DOBLADO	Project manager
Checked or approved by:	C. PLESSIS	UCM Technical project leader
Authorized by:	A. SOUBIRANE	CAD 30 UCM Product manager

Revision Record

Issue No.	Issue Date	Nature of amendment
A	12/05/2006	First version
B	12/09/2007	Document translated in English
C	18/01/2008	Take account of T_UCMHOST structure modification Add comments to structures items
D	24/06/2008	Selection first management
E	10/13/2009	Document Format New structures T_UCMHOST_DA_EPURSE_BALANCEV3, T_UCMHOST_DA_CR_EPURSE_REVALUEV3, T_UCMHOST_DA_TABLE_PRIXV3
F	01/12/2010	Document format and supplementary information
G	26/04/2010	New **T_UCMHOST_DA_PARAMV5** structure
H	30/05/2011	Structure T_UCMHOST_DA_PARAMV5. Add 'Application non response time' time-out parameter for MDB protocol. Product price displaying. See T_UCMHOST_DA_PARAM_MSGV3 field tucMsgProductSelected.

TABLE OF CONTENTS

1 INTRODUCTION

1.1 DOCUMENT PURPOSE

The purpose of this document is to describe the how develop a vending application or a DLL vending protocol.

1.2 INPUT DATA

SMO/SFO-00069	: UCM component reference manual.

1.3 TERMINOLOGY

VMC	Vending Machine Controller.
Reader	Cheap card reader terminal used for cashless payment on a vending machine.
Selection	Products distributed by a vending machine.
UCM	Universal Communication Module.
UCMC	Component UCM.

2 INTERFACE BETWEEN UCM COMPONENT AND PAIEMENTS APPLICATIONS

2.1 PRINCIPLE

When an application is called on debit_emv() or debit_non_emv() entry point, the application calls the UCM Component (UCMC) to know the treatment to be carried out (ask e-purse balance, ask debit…). Once the treatment finished the application is put on standby of a VMC or card event.

2.2 FUNCTIONS OF THE UCM COMPONENT LIBRARY

The UCM Component places at the disposal of applications a library which offers services for the management of the peripherals connected to the UCM.

This document describes the functions used to realise debit or credit payments transactions on a vending machine.

2.2.1 iLIBUCM_Pay_Ready_For_Debit

syntax	:	int iLIBUCM_Pay_Ready_For_Debit (int iSize_p, void * ps_p)
description	:	allow the payment application to ask UCMC the processing to realize on the card (e-purse debit, e-purse revalue…)
Parameters	:	iSize_p, size in bytes of the object pointed by void *. ps_p, pointer on T_UCMHOST_DEBIT structure.
Returns	:	the returned value allows the application to know which operation to realize.

UCMPAY_ASK_EPURSE_BALANCE, e-purse balance request.

UCMCPAY_SOLV, debit the e-purse request.

UCMCPAY_RECORD, record debit transaction request.

UCMPAY_CREDIT_EPURSE, revalue e-purse request.

UCMPAY_RECORD_CREDIT_EPURSE, record revalue transaction request.

UCMPAY_HOST_NOT_AVAILABLE, indicates that the VMC is not available.

UCMPAY_HOST_REMOVE_CARD, end session request.

2.2.2 iLIBUCM_Pay_Result_Debit

syntax	: int iLIBUCM_Pay_Result_Debit (int iSize_p, void * ps_p)
description	: allow the application of payment to return debit e-purse result.
parameters	: iSize_p, size in bytes of the object pointed by void *.
	ps_p, pointer on T_UCMHOST_R_DEBIT structure.
returns	: FCT_OK if e-purse is correctly debited.
	Negative value in case of error.

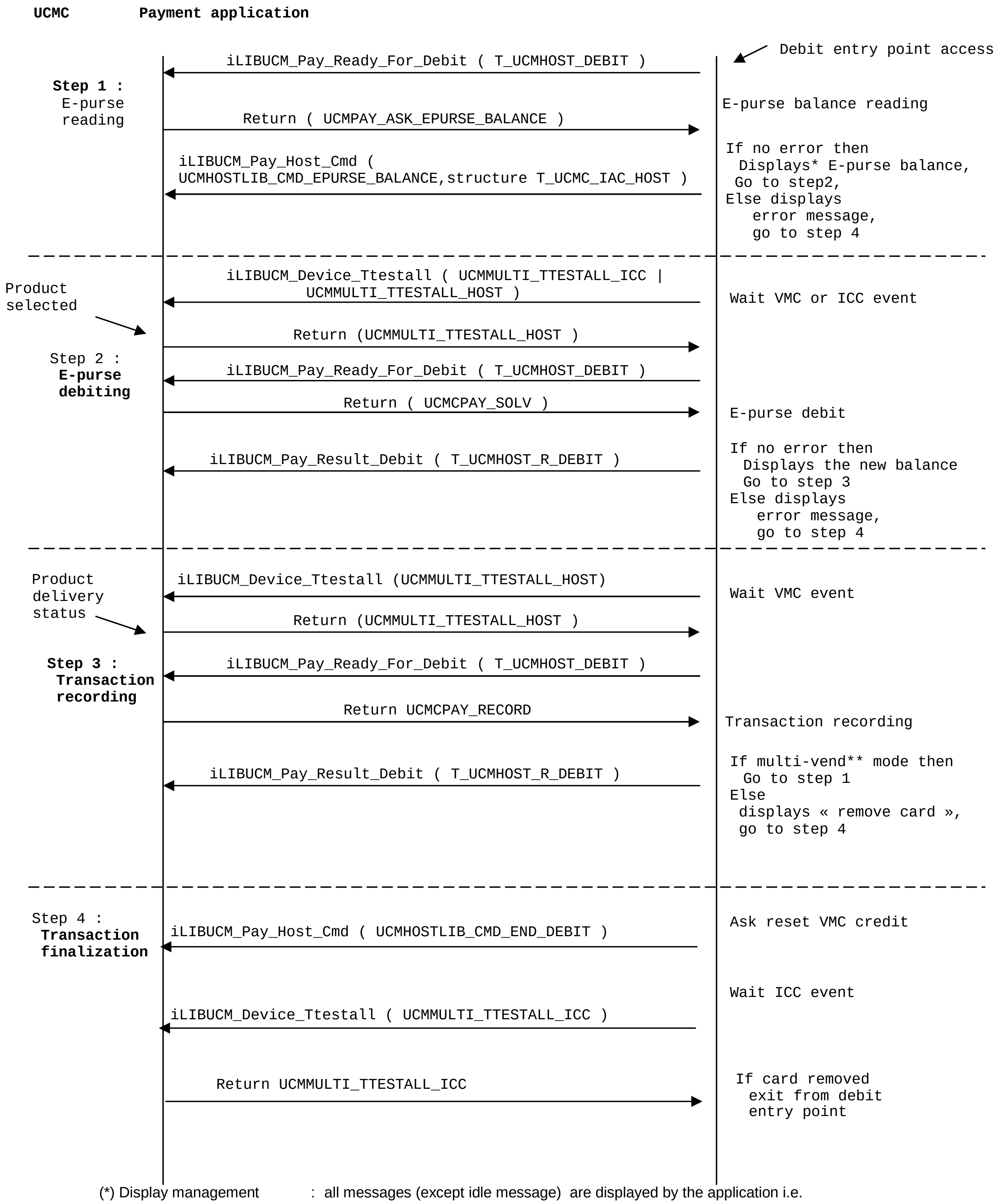
2.2.3 iLIBUCM_Pay_Host_Cmd

syntax	: int iLIBUCM_Pay_Host_Cmd (int iSize_p, void * ps_p, void * psResult_p)
description	: allow the application to send a command to the VMC via UCMC.
paramètres	: iSize_p, size in bytes of the object pointed by void *.
	ps_p, pointer on T_UCMC_IAC_HOST structure.
	psResult_p pointer on T_UCMC_IAC_HOST structure.

UCMHOSTLIB_CMD_EPURSE_BALANCE	: sends e-purse balance to the VMC.
UCMHOSTLIB_CMD_ANSW_EPURSE_REVALUE	: e-purse revalue return code.
UCMHOSTLIB_CMD_ANSW_REC_EPURSE_REVALUE	: transaction revalue record return code.
UCMHOSTLIB_CMD_PARAM_DA	: Vending machine protocol parameters.
UCMHOSTLIB_CMD_END_DEBIT	: end of session. The payment application is exit of debit entry point.

2.3 VENDING TRANSACTION FLOW

2.3.1 Vending cycle (Balance first)


UCMC
Payment application
Debit entry point access
iLIBUCM_Pay_Ready_For_Debit (T_UCMHOST_DEBIT)
Step 1 : E-purse reading
Return (UCMPAY_ASK_EPURSE_BALANCE)
E-purse balance reading
iLIBUCM_Pay_Host_Cmd (UCMHOSTLIB_CMD_EPURSE_BALANCE,structure T_UCMC_IAC_HOST)
If no error then Displays* E-purse balance, Go to step2, Else displays error message, go to step 4
Product selected
iLIBUCM_Device_Ttestall (UCMMULTI_TTESTALL_ICC | UCMMULTI_TTESTALL_HOST)
Wait VMC or ICC event
Return (UCMMULTI_TTESTALL_HOST)
Step 2 : E-purse debiting
iLIBUCM_Pay_Ready_For_Debit (T_UCMHOST_DEBIT)
Return (UCMCPAY_SOLV)
E-purse debit
iLIBUCM_Pay_Result_Debit (T_UCMHOST_R_DEBIT)
If no error then Displays the new balance Go to step 3 Else displays error message, go to step 4
Product delivery status
iLIBUCM_Device_Ttestall (UCMMULTI_TTESTALL_HOST)
Wait VMC event
Return (UCMMULTI_TTESTALL_HOST)
Step 3 : Transaction recording
iLIBUCM_Pay_Ready_For_Debit (T_UCMHOST_DEBIT)
Return UCMCPAY_RECORD
Transaction recording
iLIBUCM_Pay_Result_Debit (T_UCMHOST_R_DEBIT)
If multi-vend** mode then Go to step 1 Else displays « remove card », go to step 4
Step 4 : Transaction finalization
Ask reset VMC credit
iLIBUCM_Pay_Host_Cmd (UCMHOSTLIB_CMD_END_DEBIT)
Wait ICC event
iLIBUCM_Device_Ttestall (UCMMULTI_TTESTALL_ICC)
Return UCMMULTI_TTESTALL_ICC
If card removed exit from debit entry point

(*) Display management : all messages (except idle message) are displayed by the application i.e.

« Balance : XX.XX »
« Your choice ? »

or

« INVALID CARD »
« REMOVE CARD »

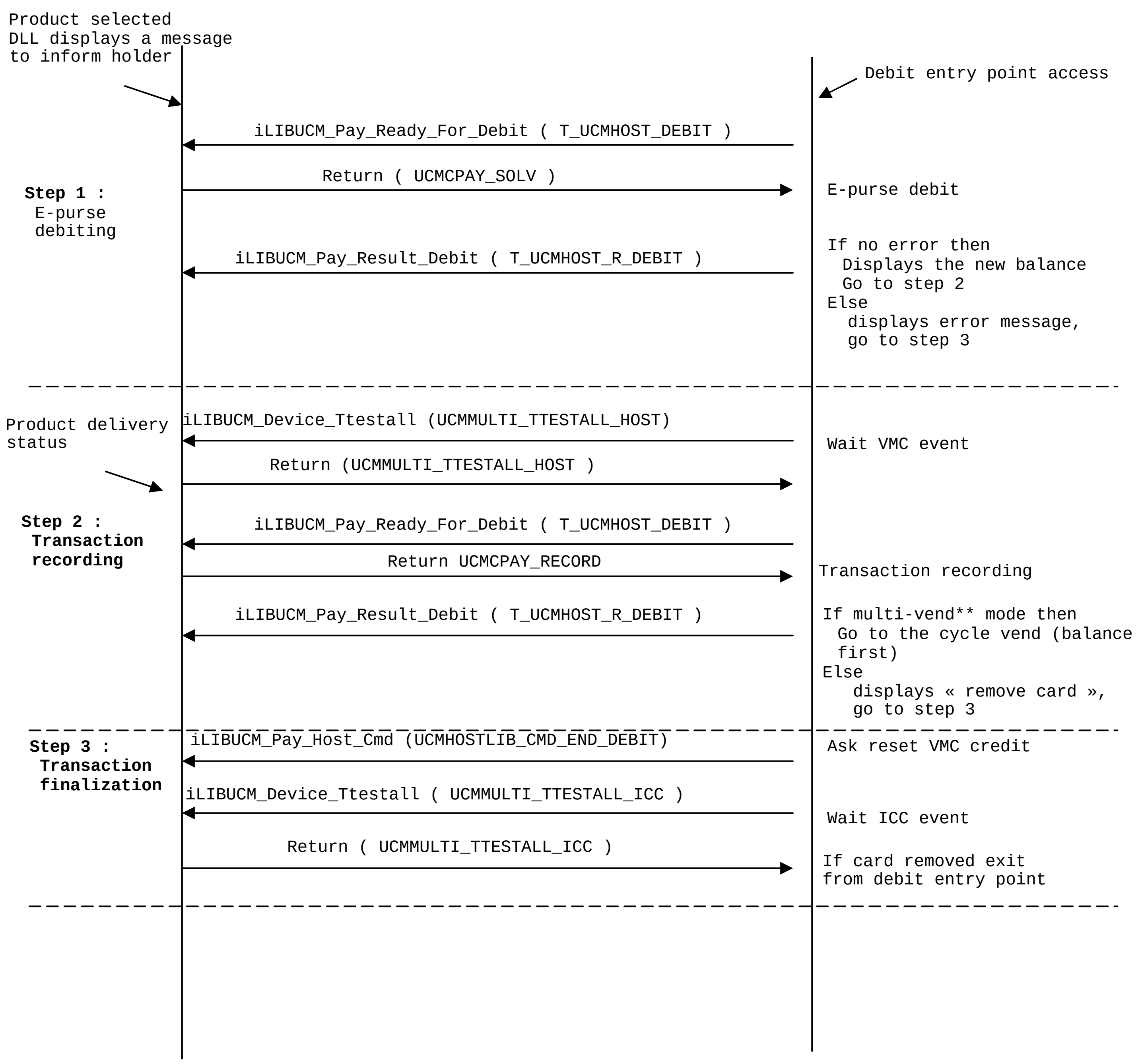
(**) Multi-vend management : when UCMC asks debit e-purse, it indicates if multi-vend is supported (c.f. T_UCMC_DA_ASK_DEBIT structure). This information is used by the application to terminate the session in case of single vend (iLIBUCM_Pay_Host_Cmd (UCMHOSTLIB_CMD_END_DEBIT) or in case of multi-vend mode to continue the session.

Note:

- After answering to succeeded debit, the Application has to wait for the message UCMCPAY_RECORD (Which is the distribution end)The application mustn't send UCMHOSTLIB_CMD_END_DEBIT between succeeded debit answer and the message UCMCPAY_RECORD. The application should display "sale in progress"
- iLIBUCM_Pay_Result_Debit (structure T_UCMHOST_R_DEBIT) has been send quickly and doesn't exceed usToWaitingCard (§3.6.8). The timeout has be used when the card has to be presented again.
- UCMMULTI_TTESTALL_ICC is use only for applications that use contact chip. For contactless applications, this event hasn't be expected.

2.3.2 Vending cycle (Selection first)

UCMC **Payment application**


Product selected
DLL displays a message to inform holder
Debit entry point access
iLIBUCM_Pay_Ready_For_Debit (T_UCMHOST_DEBIT)
Return (UCMCPAY_SOLV)
Step 1 : E-purse debiting
E-purse debit
iLIBUCM_Pay_Result_Debit (T_UCMHOST_R_DEBIT)
If no error then Displays the new balance Go to step 2 Else displays error message, go to step 3
Product delivery status
iLIBUCM_Device_Ttestall (UCMMULTI_TTESTALL_HOST)
Wait VMC event
Return (UCMMULTI_TTESTALL_HOST)
Step 2 : Transaction recording
iLIBUCM_Pay_Ready_For_Debit (T_UCMHOST_DEBIT)
Return UCMCPAY_RECORD
Transaction recording
iLIBUCM_Pay_Result_Debit (T_UCMHOST_R_DEBIT)
If multi-vend** mode then Go to the cycle vend (balance first) Else displays « remove card », go to step 3
Step 3 : Transaction finalization
iLIBUCM_Pay_Host_Cmd (UCMHOSTLIB_CMD_END_DEBIT)
Ask reset VMC credit
iLIBUCM_Device_Ttestall (UCMMULTI_TTESTALL_ICC)
Wait ICC event
Return (UCMMULTI_TTESTALL_ICC)
If card removed exit from debit entry point

(*) Display management	:	all messages (except idle message and the 1st card presentation message) are displayed by the application i.e. :

« Balance : XX.XX »

« Your choice ? »

ou

« INVALID CARD »

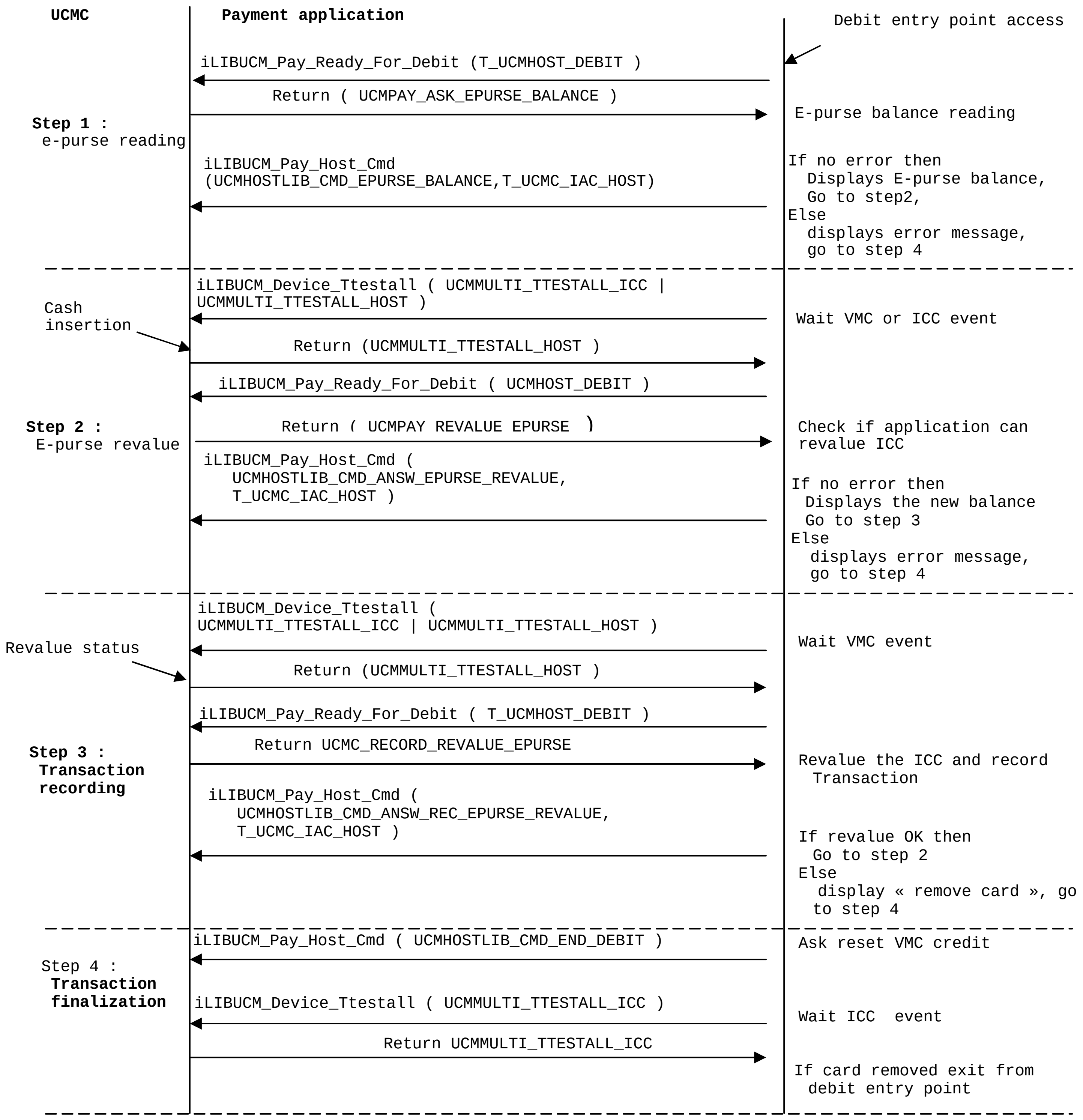
« REMOVE CARD »

(**) Multi-vend management	:	when UCMC asks debit e-purse, it indicates if multi-vend is supported (c.f. T_UCMC_DA_ASK_DEBIT structure). This information is used by the application to terminate the session in case of single vend (iLIBUCM_Pay_Host_Cmd (UCMHOSTLIB_CMD_END_DEBIT) or in case of multi-vend mode to continue the session.

Note:

- After answering to succeeded debit, the Application has to wait for the message UCMCPAY_RECORD (Which is the distribution end).
  The application mustn't to send UCMHOSTLIB_CMD_END_DEBIT between succeeded debit answer and the message UCMCPAY_RECORD. The application should display "sale in progress"

2.3.3 Revalue cycle

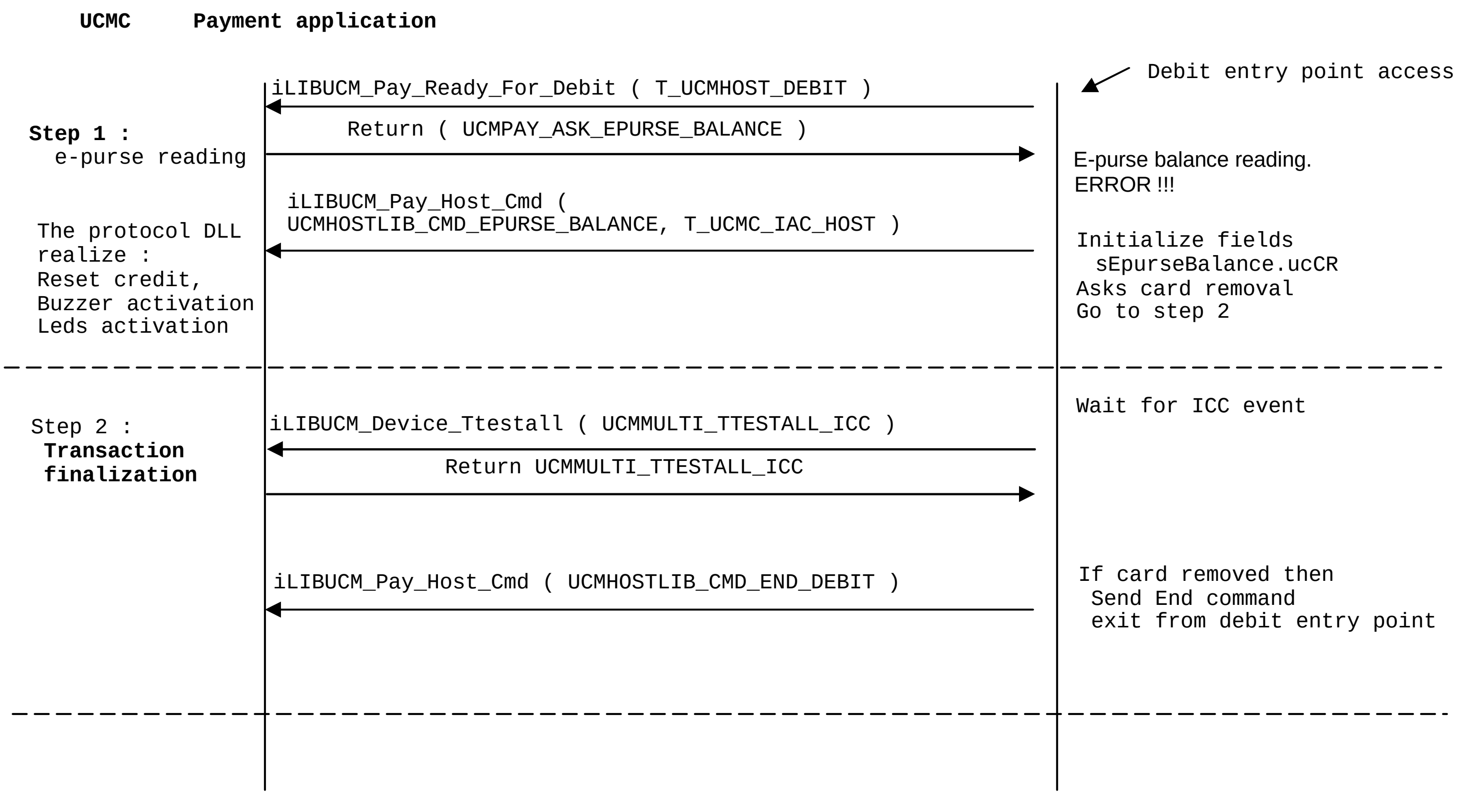

UCMC
Payment application
Debit entry point access
iLIBUCM_Pay_Ready_For_Debit (T_UCMHOST_DEBIT)
Return (UCMPAY_ASK_EPURSE_BALANCE)
E-purse balance reading
Step 1 : e-purse reading
iLIBUCM_Pay_Host_Cmd (UCMHOSTLIB_CMD_EPURSE_BALANCE,T_UCMC_IAC_HOST)
If no error then Displays E-purse balance, Go to step2, Else displays error message, go to step 4
iLIBUCM_Device_Ttestall (UCMMULTI_TTESTALL_ICC | UCMMULTI_TTESTALL_HOST)
Cash insertion
Wait VMC or ICC event
Return (UCMMULTI_TTESTALL_HOST)
iLIBUCM_Pay_Ready_For_Debit (UCMHOST_DEBIT)
Step 2 : E-purse revalue
Return (UCMPAY REVALUE EPURSE)
Check if application can revalue ICC
iLIBUCM_Pay_Host_Cmd (UCMHOSTLIB_CMD_ANSW_EPURSE_REVALUE, T_UCMC_IAC_HOST)
If no error then Displays the new balance Go to step 3 Else displays error message, go to step 4
iLIBUCM_Device_Ttestall (UCMMULTI_TTESTALL_ICC | UCMMULTI_TTESTALL_HOST)
Revalue status
Wait VMC event
Return (UCMMULTI_TTESTALL_HOST)
iLIBUCM_Pay_Ready_For_Debit (T_UCMHOST_DEBIT)
Return UCMC_RECORD_REVALUE_EPURSE
Step 3 : Transaction recording
Revalue the ICC and record Transaction
iLIBUCM_Pay_Host_Cmd (UCMHOSTLIB_CMD_ANSW_REC_EPURSE_REVALUE, T_UCMC_IAC_HOST)
If revalue OK then Go to step 2 Else display « remove card », go to step 4
iLIBUCM_Pay_Host_Cmd (UCMHOSTLIB_CMD_END_DEBIT)
Ask reset VMC credit
Step 4 : Transaction finalization
iLIBUCM_Device_Ttestall (UCMMULTI_TTESTALL_ICC)
Wait ICC event
Return UCMMULTI_TTESTALL_ICC
If card removed exit from debit entry point

Note:

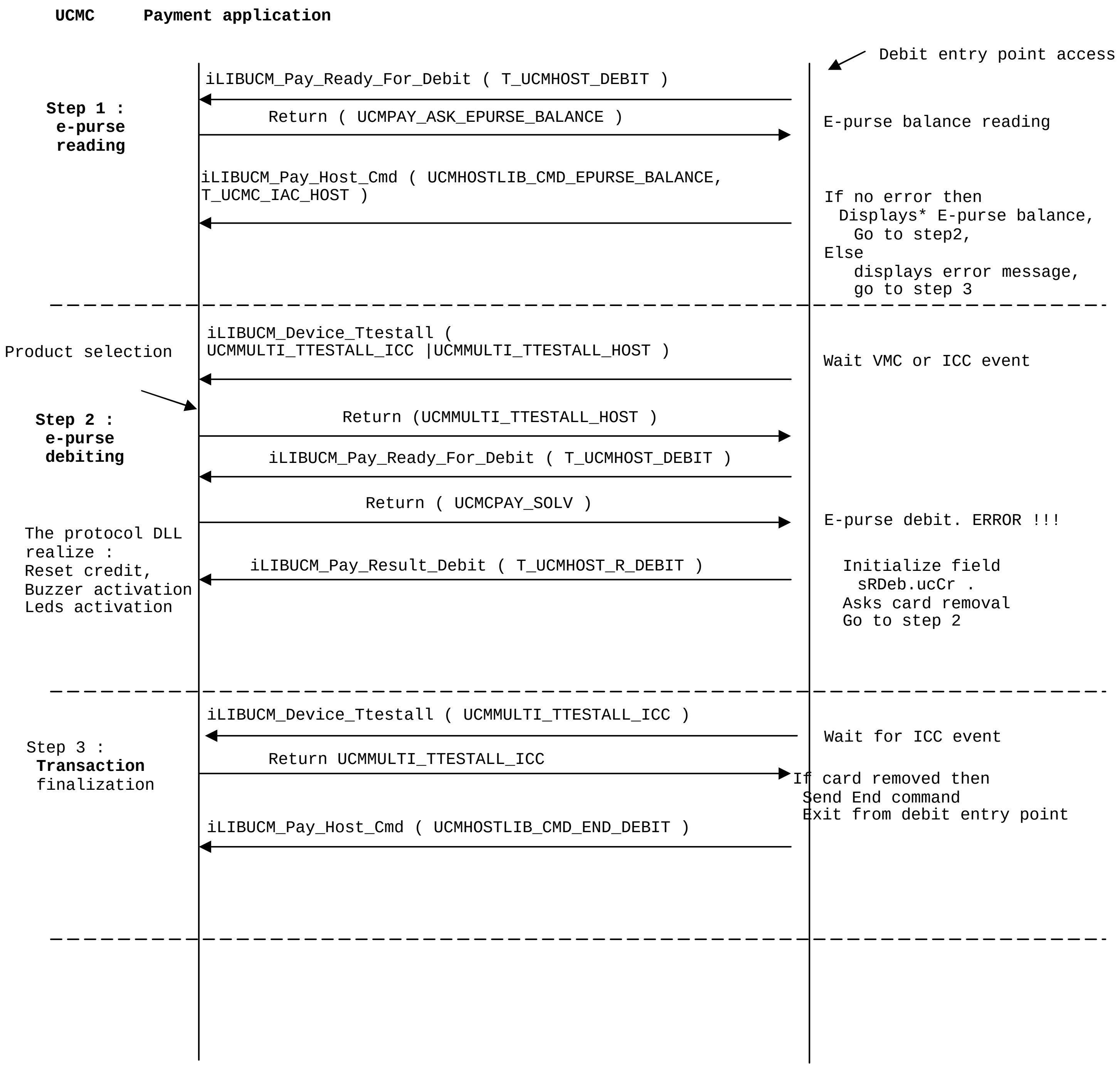
- The step 2 is used to check if the application is able to credit ICC. However, the application shouldn't credit the card. It waits for the step 3 (the coin is taken by the coin selector) to credit the card.
- UCMMULTI_TTESTALL_ICC is used only for applications that use contact chip. For contactless applications, this event hasn't be expected

2.4 ERRORS CASE

2.4.1 Card reading failure



2.4.2 E-purse debit failure


UCMC
Payment application
Debit entry point access
iLIBUCM_Pay_Ready_For_Debit (T_UCMHOST_DEBIT)
Step 1 : e-purse reading
Return (UCMPAY_ASK_EPURSE_BALANCE)
E-purse balance reading
iLIBUCM_Pay_Host_Cmd (UCMHOSTLIB_CMD_EPURSE_BALANCE, T_UCMC_IAC_HOST)
If no error then
Displays* E-purse balance,
Go to step2,
Else
displays error message,
go to step 3
Product selection
iLIBUCM_Device_Ttestall (UCMMULTI_TTESTALL_ICC |UCMMULTI_TTESTALL_HOST)
Wait VMC or ICC event
Step 2 : e-purse debiting
Return (UCMMULTI_TTESTALL_HOST)
iLIBUCM_Pay_Ready_For_Debit (T_UCMHOST_DEBIT)
Return (UCMCPAY_SOLV)
E-purse debit. ERROR !!!
The protocol DLL realize :
Reset credit,
Buzzer activation
Leds activation
iLIBUCM_Pay_Result_Debit (T_UCMHOST_R_DEBIT)
Initialize field sRDeb.ucCr .
Asks card removal
Go to step 2
iLIBUCM_Device_Ttestall (UCMMULTI_TTESTALL_ICC)
Wait for ICC event
Step 3 : Transaction finalization
Return UCMMULTI_TTESTALL_ICC
If card removed then
Send End command
Exit from debit entry point
iLIBUCM_Pay_Host_Cmd (UCMHOSTLIB_CMD_END_DEBIT)

2.5 STRUCTURES USED

2.6 T_UCMHOST_DEBIT STRUCTURE

```
typedef struct
{
  unsigned long   ulAmount ;        /* NA */
  S_MONEY         tCurrency;        /* NA */
  unsigned char   ucTrsType;        /* NA */
  unsigned char   ucTrsEntry;       /* NA */
  unsigned char   ucTrsMode;        /* NA */
  unsigned char   ucTrsSupport;     /* NA */
  unsigned char   ucFunction;       /* UCMHOST_FCT_SOLV, UCMHOST_FCT_ENREG, UCMHOST_FCT_REVALUE */
unsigned char   ucMode;    /* NA */
unsigned char   ucClasse;  /* NA */
unsigned char   ucPrint;   /* NA */
unsigned char   ucDisplay;          /* NA */
unsigned short  usToWaitingCard;    /* NA */
unsigned short  usToRemovedCard;    /* NA */
unsigned char   ucAppliNum;         /* Application number if specific */
unsigned char   ucPowerOn; /* 1= Power on for solv */
union
{
  unsigned char            ucRuf[ 40 ];    /* Reserve */
  T_UCMC_DA_ASK_DEBIT      tDaAskForDebit ;/* UCM-> App */
  T_UCMC_DA_CR_DISTRIBUTIONtDaCrDistribution ;   /* UCM-> App */
  T_UCMC_DA_ASK_REVALUE    tDaAskForRevalue ;    /* UCM-> App */
  T_UCMC_DA_REC_EPURSE_REVALUE      tDARecRevalue ;          /* UCM-> App */
  T_UCMHOST_SOLV_COMP_LOC  tSolvLoc ;      /* not used  */
} u ;

} T_UCMHOST_DEBIT ;
```

This structure uses sub-structures T_UCMC_DA_ASK_DEBIT, T_UCMC_DA_CR_DISTRIBUTION, T_UCMC_DA_ASK_REVALUE, T_UCMC_DA_CR_REVALUE.
It's ucFunction field which allows to know the sub-structure to use. Moreover, each sub-structure has a field ucCmd which corresponds to the command to execute.

4 commands are defined :

- UCMCPAY_SOLV to request an e-purse debit to the application,
- UCMC_PAYRECORD to request to record a debit transaction (distribution return code),
- UCMC_REVALUE_EPURSE to request an e-purse revalue,
- UCMC_RECORD_REVALUE_EPURSE to request to record a revalue transaction.

2.6.1 UCMCPAY_SOLV command

```
typedef struct
{
  unsigned char             ucCmd  ;                 /* UCMCPAY_SOLV */
  unsigned char             ucSelectionNumber ;      /* 0 <= n° selection < 99 */
  unsigned char             ucVendType ;             /* multi-vend is possible ? */
  unsigned char             ucSelectionNotDefined ;  /* 1 = not defined,
                                                        0xff = price holding */
  unsigned int              uiSelectionPrice ;
  unsigned char             tucCurrencyCode [ 3 ] ;
} T_UCMHOST_DA_ASK_DEBIT ;
```

2.6.2 Command UCMC_PAYRECORD

```
typedef struct
{
unsigned char               ucCmd ;                         /* UCMCPAY_RECORD */
unsigned char               ucCrDistribution ;
unsigned char               ucSelectionNumber ;
union
} T_UCMHOST_DA_CR_DISTRIBUTION ;
```

2.6.3 Command UCMC_REVALUE_EPURSE

```
typedef struct
{
unsigned char               ucCmd ;                  /* UCMC_REVALUE_EPURSE */
unsigned char               tucCurrencyCode [ 3 ] ;
unsigned int                uiRevalueAmount ;
} T_UCMC_DA_ASK_REVALUE ;
```

2.6.4 Command UCMC_RECORD_REVALUE_EPURSE

```
typedef struct
{
   unsigned char ucCrRevalue ;                              /*UCMHOST_CR_OK*/
   unsigned char tucRuf [ 3 ]
   unsigned long ulRevalueAmount ;
   unsigned long ulEpurseBalance ;
}T_UCMHOST_DA_CR_EPURSE_REVALUEV3 ;
```

2.7 STRUCTURE T_UCM_IAC_HOST

```
typedef struct
{
 unsigned short      usHostCmd ;
 unsigned char       ucCr ;
 unsigned char       ucRuf ;
 unsigned short      usHostWaitTimeout ;       /* second */
 unsigned short      usSize ;     /* of data in union */
 union
 {
   unsigned char         ucBuf [ UCMC_IAC_HOST_BUFFER_SIZE ] ;
   T_UCMHOST_DA_EPURSE_BALANCEV3  tEpurseBalance ;
   T_UCMHOST_DA_CR_EPURSE_REVALUEV3    tEpurseRevalue ;
   T_UCMHOST_DA_REC_EPURSE_REVALUEtRecEpurseRevalue ;
   T_UCMHOST_DA_PARAMV3  tParamDa ;
 } u;
} T_UCMC_IAC_HOST;
```

The field usHostCmd determines the command and the union structure to use. The authorised commands are defined below.

2.7.1 UCMHOSTLIB_CMD_EPURSE_BALANCE command

This command uses T_UCMC_DA_EPURSE_BALANCEV3 sub-structure.

```
typedef struct
{
 unsigned long           ulEpurseBalance ;
 unsigned char           ucCr ;                /*UCMHOST_CR_OK*/
 unsigned char           tucCurrencyCode [ 3 ] ;
 unsigned char           tucLanguageCode [ 3 ] ;
 unsigned char           ucAllowRevalue ;
 unsigned char           ucAllowRefund ;
 unsigned char           ucAllowDisplayBalance ;
 unsigned char           ucAllowMultiVend ;
 unsigned char           ucRuf ;  // if ucAllowRevalue=TRUE else set to 0
 unsigned long           ulRevalueLimitBalance ;  // max value of Epurse Balance(*)
 unsigned long           ulRevalueLimitAmount ;   // max value of coins used for revalue(*)
                                                     (coins upper limit)
} T_UCMHOST_DA_EPURSE_BALANCEV3;
```

 (*) : must be initialized if ucAllowRevalue=TRUE. if ucAllowRevalue=FALSE must be set to 0.

2.7.2 UCMHOSTLIB_CMD_ANSW_EPURSE_REVALUE command

This command uses T_UCMC_DA_EPURSE_REVALUE sub-structure.

```
typedef struct
{
   unsigned char ucCrRevalue ;          /*UCMHOST_CR_OK*/
   unsigned char tucRuf [ 3 ] ;         /* v0200 size = 3 instead of 2 */
   unsigned long ulRevalueAmount ;
   unsigned long ulEpurseBalance ;
}T_UCMHOST_DA_CR_EPURSE_REVALUEV3 ;
```

2.7.3 UCMHOSTLIB_CMD_ANSW_REC_EPURSE_REVALUE command

This command uses T_UCMC_DA_REC_EPURSE_REVALUE sub-structure.

```
typedef struct
{
   unsigned char          ucCmd ;
   unsigned char          ucType ;
   unsigned char          tucRuf ;
   unsigned char          ucCrRecRevalue ;         /* UCMHOST_CR_OK if OK */

} T_UCMC_DA_REC_EPURSE_REVALUE ;
```

2.7.4 UCMHOSTLIB_CMD_PARAM_DA command

This command uses T_UCMHOST_DA_PARAMV4 sub-struct.

This structure is detailed in the section « T_UCMHOST_DA_PARAMV4 structure».

2.7.5 UCMHOSTLIB_CMD_END_DEBIT

This command doesn't have sub-struct.

2.8 STRUCTURE T_UCMHOST_R_DEBIT_DA

```
typedef struct
{
         unsigned char            ucCr;                  /* Debit CR. UCMHOST_CR_OK = OK */
         unsigned char            ucDiag ;               /* return code if error */
         unsigned char            ucUCMDiag ;            /* for UCMC: 0 = OK,
                                                            1 = Service not called,
                                                            2 = Called service returned KO,
                                                            3 = No appli*/
         unsigned char            ucPrinter;             /* NA */
         unsigned char            ucDisplay;             /* RUF */
         unsigned char            ucCardInside;          /* 1=Card inside during transaction*/
         unsigned char            ucMode;                /* NA */
         unsigned char            ucFunction ;           /* UCMHOST_FCT_SOLV,
                                                            UCMHOST_FCT_ENREG
                                                            UCMHOST_FCT_REVALUE */
         unsigned char            ucTypeCardStruct ;     /* NA */
         unsigned char            ucSupport ;            /* NA */
         unsigned short           usAppName ;            /* application segment
                                                            number*/
         T_AFFNOM                 tAppLibelle;           /* application name
                                                            i.e CLIPURSE */
         MONTANT                  ulAmount;              /* transaction amount */
         S_MONEY                  tCurrency;             /* currency of the transaction */
         unsigned char            ucCardHolderLanguage;  /* NA */
         union
         {
            unsigned char         ucBuf[ 20 ] ;
            T_UCMHOST_SOLV_COMP_LOC  sLoc;               /* NA */
         } uRuf;

         union
         {
            T_UCMHOST_CARD        sCard ;
            T_UCMHOST_CARD_MONEO  sCardMoneo;
            T_UCMHOST_R_DEBIT_DA  sRDebitDa ;
            unsigned char         ucBuf [  UCMHOST_MAX_SIZE_CARD_APPLI_INFO +
                                           UCMHOST_MAX_SIZE_CARD_INFO +
                                           UCMHOST_MAX_SIZE_CARD_ACCEPT_INFO ] ;
         }u;
      } T_UCMHOST_R_DEBIT ;


typedef struct
{
         MONTANT                  ulEpurseBalance ;
      } T_UCMHOST_R_DEBIT_DA ;
```

3 INTERFACE BETWEEN THE UCM COMPONANT AND A PROTOCOL DLL

3.1 PRINCIPLE

The protocol DLL manages 2 buffers (fifo). Once for the messages received from the VMC. The other one for messages received from UCMC. Two functions allow to read and to send messages.

3.2 PROTOCOLE DLL FUNCTION

3.2.1 iUcmHostDll_Read_Msg

syntax	:	int iUcmHostDll_Read_Msg (T_UCMHOST psMessage_p)
description	:	allows UCMC to read a message in the fifo.
Parameter	:	psMessage_p, pointer on T_UCMHOST_DEBIT structure.
Return	:	FCT_OK if function correctly executed else return negative value.

3.2.2 iUcmHostDll_Send_Msg

syntax	:	int iUcmHostDll_Send_Msg (T_UCMHOST psMessage_p)
description	:	allows UCMC to write a message in the fifo.
Parameter	:	psMessage_p, pointer on T_UCMHOST_DEBIT structure.
Return	:	FCT_OK if function correctly executed else return negative value.

3.2.3 iUCMHOST_Give_Status

syntax	:	int iUcmHostDll_Give_Status (T_UCMHOST_STATUS psHostStatus_p)
description	:	allows UCMC to get protocol DLL status.
Parameter	:	psHostStatus _p, pointer on T_UCMHOST_STATUS structure.
Return	:	FCT_OK if function correctly executed else return negative value.

3.2.4 Exchanged messages

3.2.4.1 Messages from protocol DLL to UCM component

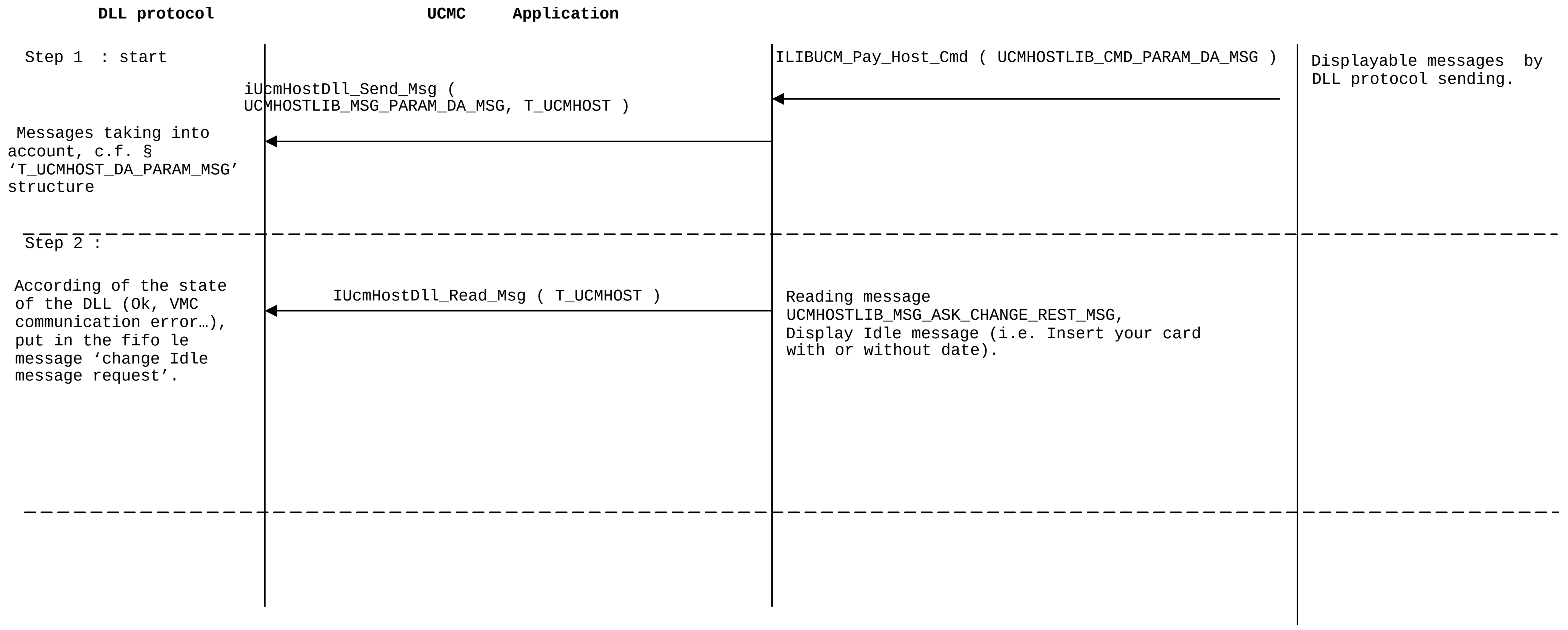
UCMHOSTLIB_MSG_ASK_CHANGE_REST_MSG	:	change Idle message request.
UCMHOSTLIB_MSG_ASK_REMOVE_CARD	:	close the session request.
UCMHOSTLIB_MSG_ASK_DEBIT	:	e-purse debit request.
UCMHOSTLIB_MSG_CR_DISTRIBUTION	:	product distribution report.
UCMHOSTLIB_MSG_ASK_REVALUE	:	e-purse revalue request.
UCMHOSTLIB_MSG_REC_REVALUE	:	request to record last revalue e-purse.
UCMHOSTLIB_MSG_ASK_DISP_MSG_APPLI	:	request to send displayed messages to protocol DLL.

3.2.4.2 Messages from UCM component to protocol DLL

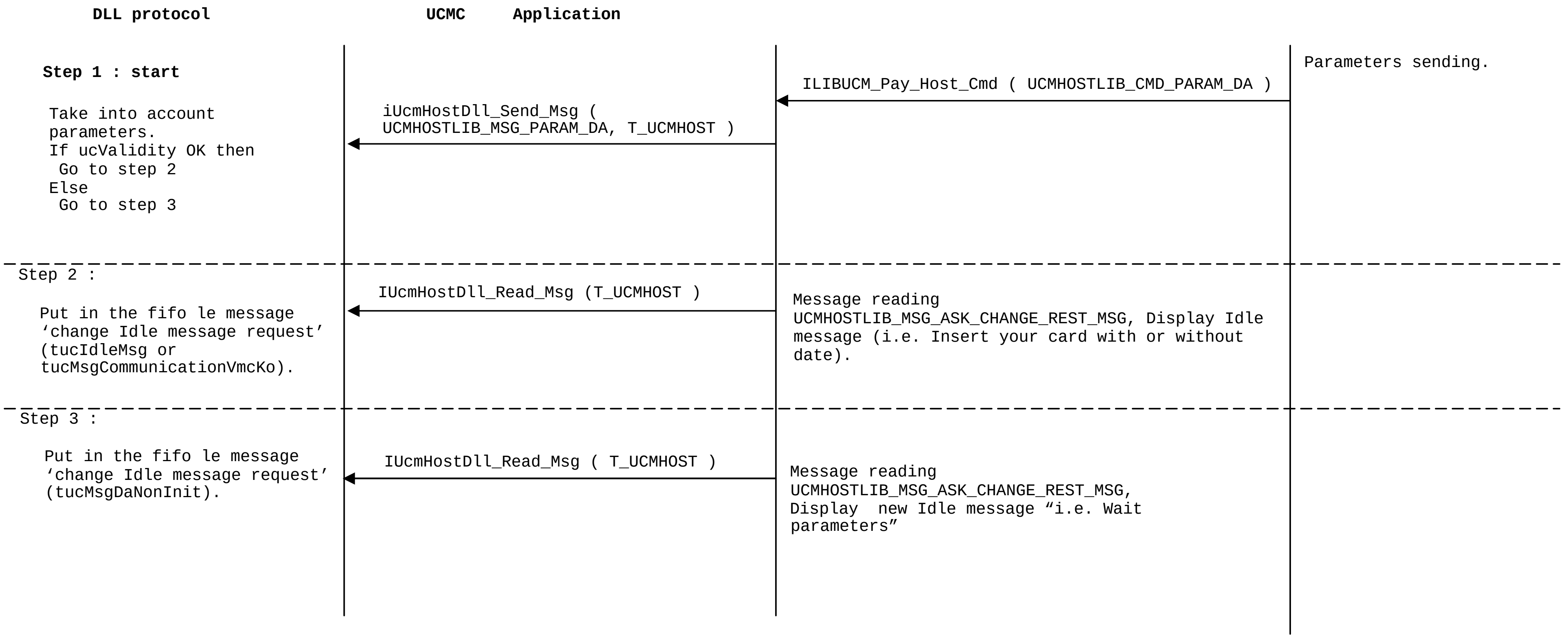
UCMHOSTLIB_MSG_PARAM_DA	:	protocol DLL parameter setting request.
UCMHOSTLIB_MSG_PARAM_DA_MSG	:	displayed messages by protocol DLL.
UCMHOSTLIB_MSG_EPURSE_BALANCE	:	e-purse balance.
UCMHOSTLIB_MSG_ANSW_DEBIT	:	answer to e-purse debit request.
UCMHOSTLIB_MSG_ANSW_REVALUE	:	answer to e-purse revalue request.
UCMHOSTLIB_MSG_END	:	reset VMC credit request.

3.3 DLL PROTOCOLE PARAMETER SETTINGS

3.3.1 Messages parameter setting


DLL protocol
UCMC
Application
Step 1 : start
ILIBUCM_Pay_Host_Cmd (UCMHOSTLIB_CMD_PARAM_DA_MSG)
Displayable messages by DLL protocol sending.
iUcmHostDll_Send_Msg (UCMHOSTLIB_MSG_PARAM_DA_MSG, T_UCMHOST)
Messages taking into account, c.f. § ‘T_UCMHOST_DA_PARAM_MSG’ structure
Step 2 :
According of the state of the DLL (Ok, VMC communication error…), put in the fifo le message ‘change Idle message request’.
IUcmHostDll_Read_Msg (T_UCMHOST)
Reading message UCMHOSTLIB_MSG_ASK_CHANGE_REST_MSG, Display Idle message (i.e. Insert your card with or without date).

3.3.2 DLL protocol parameter setting



DLL protocol **UCMC** **Application**

Parameters sending.

ILIBUCM_Pay_Host_Cmd (UCMHOSTLIB_CMD_PARAM_DA)

Step 1 : start

Take into account parameters.
If ucValidity OK then
 Go to step 2
Else
 Go to step 3

iUcmHostDll_Send_Msg (UCMHOSTLIB_MSG_PARAM_DA, T_UCMHOST)

Step 2 :

Put in the fifo le message 'change Idle message request' (tucIdleMsg or tucMsgCommunicationVmcKo).

IUcmHostDll_Read_Msg (T_UCMHOST)

Message reading UCMHOSTLIB_MSG_ASK_CHANGE_REST_MSG, Display Idle message (i.e. Insert your card with or without date).

Step 3 :

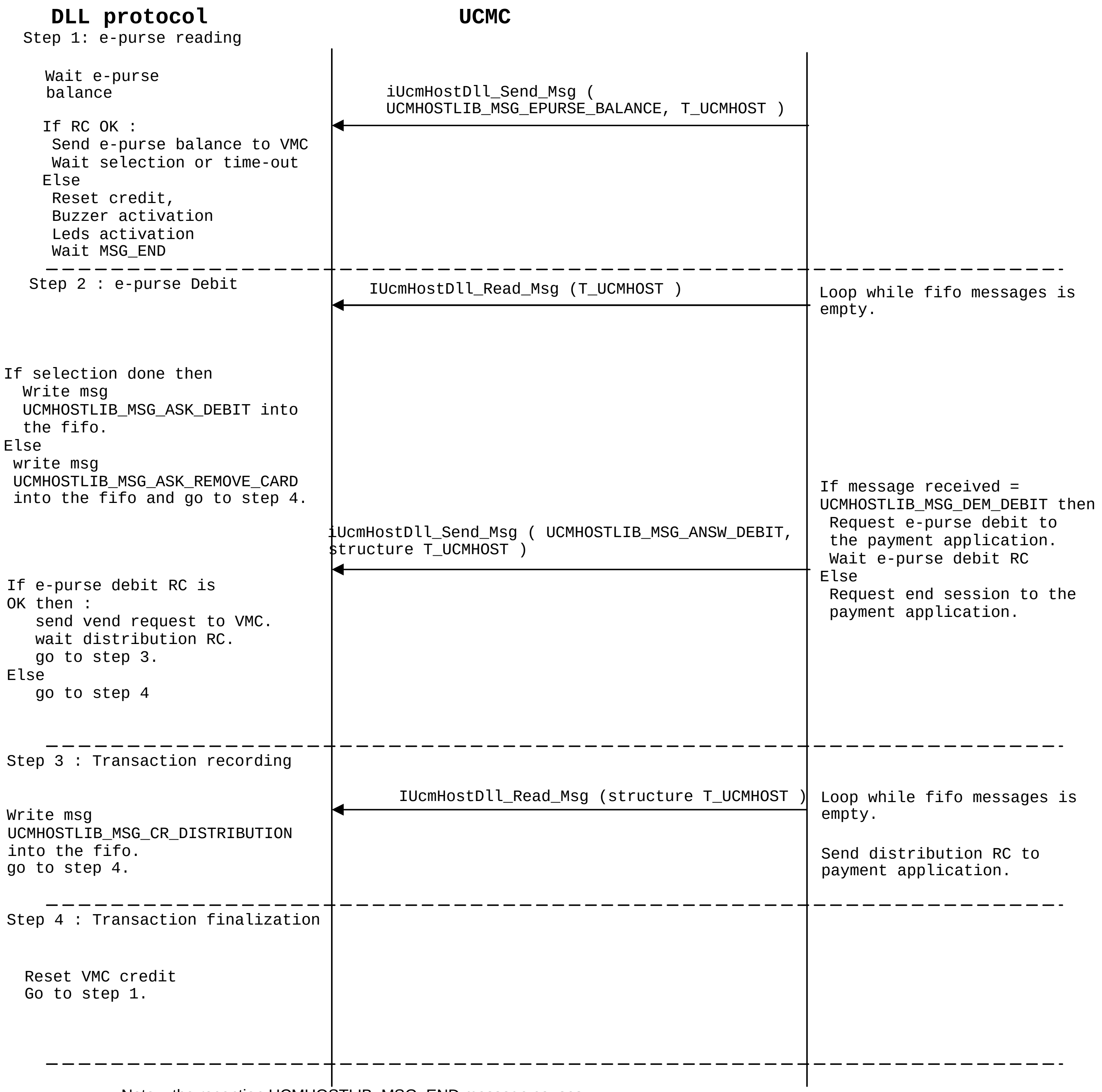
Put in the fifo le message 'change Idle message request' (tucMsgDaNonInit).

IUcmHostDll_Read_Msg (T_UCMHOST)

Message reading UCMHOSTLIB_MSG_ASK_CHANGE_REST_MSG, Display new Idle message "i.e. Wait parameters"

3.4 TRANSACTION FLOW

3.4.1 Vend cycle



DLL protocol | **UCMC**

```
Step 1: e-purse reading

  Wait e-purse
  balance                 iUcmHostDll_Send_Msg (
                          UCMHOSTLIB_MSG_EPURSE_BALANCE, T_UCMHOST )
  If RC OK :
   Send e-purse balance to VMC
   Wait selection or time-out
  Else
   Reset credit,
   Buzzer activation
   Leds activation
   Wait MSG_END
```

```
Step 2 : e-purse Debit    IUcmHostDll_Read_Msg (T_UCMHOST )      Loop while fifo messages is
                                                                 empty.

If selection done then
  Write msg
  UCMHOSTLIB_MSG_ASK_DEBIT into
  the fifo.
Else
 write msg
 UCMHOSTLIB_MSG_ASK_REMOVE_CARD                                  If message received =
 into the fifo and go to step 4.                                 UCMHOSTLIB_MSG_DEM_DEBIT then
                                                                  Request e-purse debit to
                          iUcmHostDll_Send_Msg ( UCMHOSTLIB_MSG_ANSW_DEBIT,   the payment application.
                          structure T_UCMHOST )                   Wait e-purse debit RC
                                                                 Else
If e-purse debit RC is                                            Request end session to the
OK then :                                                         payment application.
   send vend request to VMC.
   wait distribution RC.
   go to step 3.
Else
   go to step 4
```

```
Step 3 : Transaction recording

                          IUcmHostDll_Read_Msg (structure T_UCMHOST )   Loop while fifo messages is
Write msg                                                               empty.
UCMHOSTLIB_MSG_CR_DISTRIBUTION
into the fifo.                                                          Send distribution RC to
go to step 4.                                                           payment application.
```

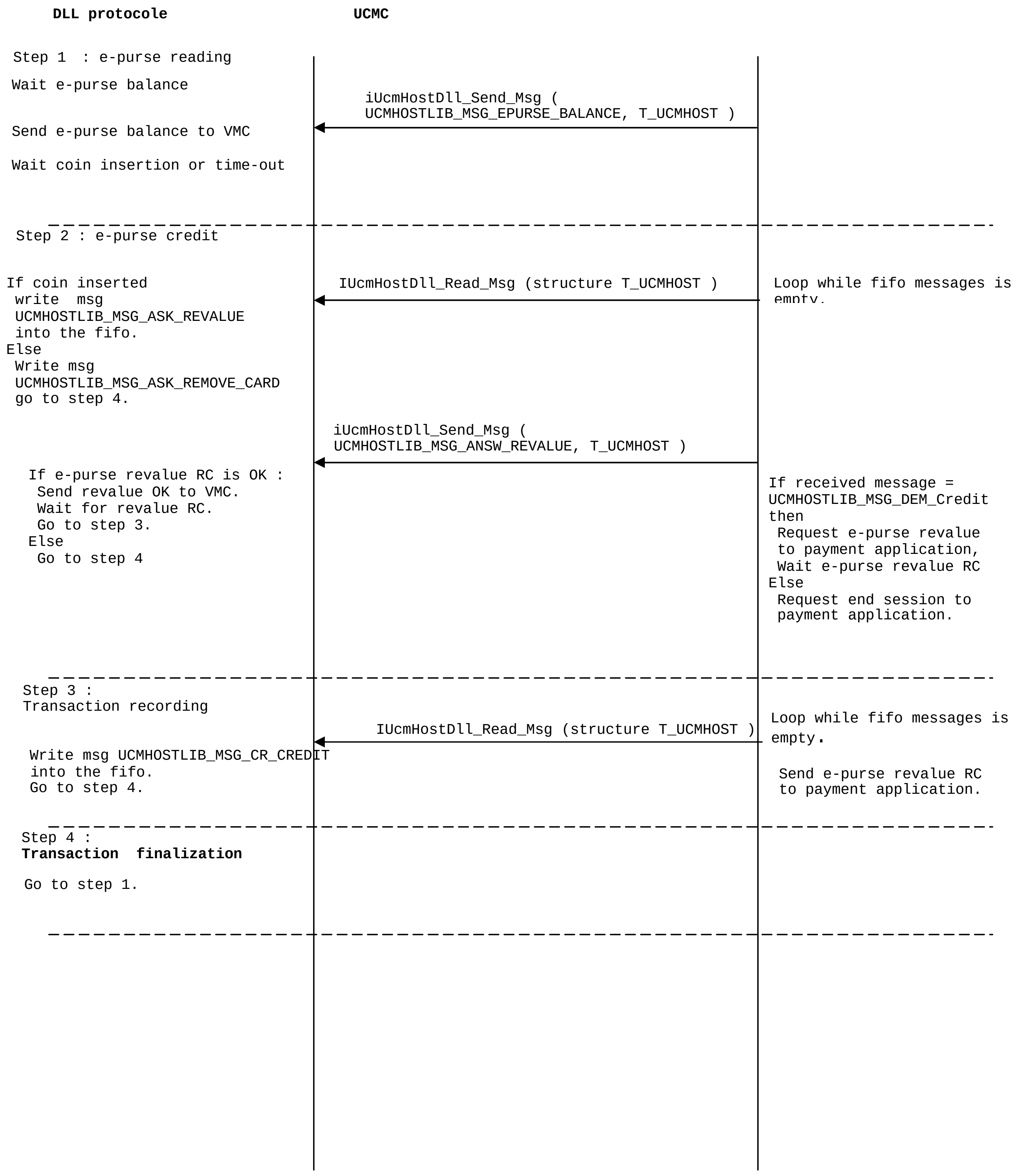
```
Step 4 : Transaction finalization

  Reset VMC credit
  Go to step 1.
```

Note : the reception UCMHOSTLIB_MSG_END message causes :

- VMC credit reset
- Come back to step 1.

3.4.2 Revalue cycle



DLL protocole | **UCMC**

Step 1 : e-purse reading

Wait e-purse balance

iUcmHostDll_Send_Msg (UCMHOSTLIB_MSG_EPURSE_BALANCE, T_UCMHOST)

Send e-purse balance to VMC

Wait coin insertion or time-out

Step 2 : e-purse credit

If coin inserted
 write msg
 UCMHOSTLIB_MSG_ASK_REVALUE
 into the fifo.
Else
 Write msg
 UCMHOSTLIB_MSG_ASK_REMOVE_CARD
 go to step 4.

IUcmHostDll_Read_Msg (structure T_UCMHOST)

Loop while fifo messages is empty.

iUcmHostDll_Send_Msg (UCMHOSTLIB_MSG_ANSW_REVALUE, T_UCMHOST)

If e-purse revalue RC is OK :
 Send revalue OK to VMC.
 Wait for revalue RC.
 Go to step 3.
Else
 Go to step 4

If received message = UCMHOSTLIB_MSG_DEM_Credit then
 Request e-purse revalue to payment application,
 Wait e-purse revalue RC
Else
 Request end session to payment application.

Step 3 :
Transaction recording

IUcmHostDll_Read_Msg (structure T_UCMHOST)

Loop while fifo messages is empty.

Write msg UCMHOSTLIB_MSG_CR_CREDIT
into the fifo.
Go to step 4.

Send e-purse revalue RC to payment application.

Step 4 :
Transaction finalization

Go to step 1.

3.5 STRUCTURES USED

3.6 PRINCIPLE

Each message uses T_UCMHOST structure. The structure fields are function of the type of message (union). Below is described the T_UCMHOST message according with the message type.

3.6.1 T_UCMHOST structure

It's the structure used to exchange data between UCMC and protocol DLL.

```
typedef struct
{
   unsigned short                        usWho;           /* NA */
   unsigned short                        usType;          /* message type */
   int                                   iStatus;         /* message status */
   unsigned int                          uiNbApp;         /* NA */
   unsigned int                          uiSize;          /* area size pointed by u */
   union
   {
    unsigned char                        *  pucData ;
    void                                 *  pvData ;
    void                                 ** ppvData ;
    T_UCMHOST_DEBIT                      *  psDebit ;
    T_UCMHOST_R_DEBIT                    *  psDebit_R ;
    T_UCMHOST_STATUS_UCM                 *  pUCMStatus ;
    T_UCMHOST_DEM_FCTAPP                 *  pDFctApp ;
    T_UCMHOST_FCTAPP                     *  pFctApp ;
    T_UCMHOST_D_CONNECT                  *  pDConnect ;
    T_UCMHOST_R_CONNECT                  *  pConnect_R ;
    T_UCMHOST_CNX_READ                   *  pCnxRead;
    T_UCMHOST_SPEED_DIAL                 *  pSpeed ;
    T_UCMHOST_R_SPEED_DIAL               *  pSpeed_R ;
    T_UCMHOST_R_CANCEL                   *  pCancel ;
    T_UCMHOST_CONSO                      *  pConsol ;
    T_UCMHOST_NEW_DATE                   *  pDate ;
    T_UCMHOST_R_NEW_DATE                 *  pDate_R ;
    T_UCMHOST_R_MTNC                     *  pRMtnc ;
    T_UCMHOST_MSG_DISPLAY                *  pDisplay;
    T_UCMHOST_ASK_MSG_DISPLAY            *  pDisplayAsk;
    T_UCMHOST_DEVICE_CONF                *  pConfDevice;
    T_UCMHOST_DA_PARAMV4                 *  pParamDa ;
    T_UCMHOST_DA_PARAM_MSGV3             *  pParamDaMsg ;
    T_UCMHOST_DA_EPURSE_BALANCEV3        *  pEPurseBal ;
    T_UCMHOST_DA_CR_EPURSE_REVALUEV3     *  pCrRevalue ;
    T_UCMHOST_DA_CR_REC_EPURSE_REVALUE   *  pCrRecRevalue ;
    unsigned int                         *  puiReason ;
    T_UCMHOST_APP_MSG                    *  pAppMgs ;
    T_UCMHOST_HOST_DATA                  *  pHostData ;
   }u;
}T_UCMHOST ;
```

3.6.2 T_UCMC_DA_PARAMV5 structure

This structure contain parameters necessary to initialise protocol DLL.

```
typedef struct
{
   unsigned char            ucValidity;                    /* (1) */
   unsigned char            tucTerminalNumber [ 10 + 1 ] ; /* (2) */
   unsigned char            ucVmcType [ 2 ] ;              /* (3) */
   unsigned short           usiTimeOutIfNoSelection ;      /* (4) */
   unsigned short           usiTimeOutBuzzer ;             /* (5) */
   unsigned short           usiBuzzerDuration ;            /* (6) */
   unsigned char            ucDigitNumber ;                /* (7) */
   unsigned char            tucCurrencyLabel [ 3 + 1 ] ;   /* (8) */
   unsigned char            tucCurrencyCode [ 3 + 1 ] ;    /* (9) */
   unsigned char            ucMultivendPossible;           /* (10) */
   unsigned char            ucPriceHolding;                /* (11) */
   unsigned char            ucRuf;
   unsigned int             usiScaleFactor;                /* (12) */
   unsigned char            ucNbSelection;                 /* (13) */

   unsigned char            ucRuf2 ;
   unsigned short int       usiTimeOutVM ;                 /* (14) */
   T_UCMHOST_DA_TABLE_PRIX  tPriceTable [ 100 ] ;          /* (15) */
   unsigned short           usTimeOutIfSelected ;          /* (16) */
   unsigned char            ucVendingMode ;                /* (17) */
   unsigned char            ucDisplayPrice;                /* (18) */
   unsigned int             uiDllParameters ;              /* (19) */
   unsigned short int       usiCashLessAdress ;            /* (20) */
   unsigned char            ucDisplayDa;                   /* (21) */
   unsigned char            ucRuf3 ;                       /* (22) new */
}T_UCMHOST_DA_PARAMV5 ;
```

(1) : 1 → Communication with VMC is ON
0 → Communication with VMC is OFF (The following data are not read)

(2) : Terminal number . (for ex "0000001234")

(3) : VMC type. RFU. Set to {0,0}

(4) : Time out (second), before activing buzzer if a card is inserted and no selection is done.

(5) : Time out (second), before activing buzzer after having selected a product.

(6) : Buzzer duration (second).

(7) : Number of digit after comma.

(8) : Currency label (c.f. ISO 4217). For ex "EUR"

(9) : Currency code (c.f. ISO 4217). For ex "978"

(10) : Type of vend (Single vend / multi-vend).
0→ the vend mode is single
1→ the vend mode is multiple

(11) : Price calculation mode.
0→The price is provided by the VMC (Master). The price is multiplied by the scale factor.
1→ The price is set by the CAD30. The price is set in the tPriceTable (by using selection number)

(12) : Scale Factor used if the price is provided by the VMC. → Generally, set to 1

(13) : Number of selections available on VMC.

(14) : Time out (second) Dll protocol waits VM distribution result. By default must be set to 90.

(15) : Selections price array. The cash and cashless prices can be defined for 100 selections (0 to 64).

(16) : Time out (second) to present a card when a product is selected (contactless) >0 only if Vending mode != UCMHOST_VENDING_MODE_BALANCE_FIRST

(17) : Vending mode
UCMHOST_VENDING_MODE_BALANCE_FIRST (default mode)→ The balance is sent to VMC and the application waits for selection.
UCMHOST_VENDING_MODE_SELECTION_FIRST→ A product can be selected form idle state without having sent the balance.
UCMHOST_VENDING_MODE_PUSHBUTTON (not available)

(18) : Available only with 2EXE configuration
UCMHOST_DISPLAY_PRICE_NONE → No price is displayed on VMC
UCMHOST_DISPLAY_PRICE_CASH → The Cash price (uiPrixEspece §3.6.3) is displayed on VMC when a product is selected from Idle state
UCMHOST_DISPLAY_PRICE_CARD → The Card price (uiPrixCarte §3.6.3) is displayed on VMC when a product is selected from Idle state.
Either → set to 0

(19) : Bits field where
b0 = ACTIVE GATEWAY. Allows to activate GateWay communication to retrieve audit data from VM.
b1 = ACTIVE SIELAFF MODE
b2 = MDB ANSWER MODE => 1 = direct to a command, 0 = wait poll cmd to answer. We recommend to set this bit to 0.

(20)	:	Available only with MDB configuration. Cashless reader peripheral address. By default must be set to 0x10.
(21)	:	use Vending Machine display
(23)	:	Application time-out response value (only for MDB protocole). For a time-out of 25 second ucRuf3=25. Time-out value must be ranging between 5 and 255.

3.6.3 T_UCMHOST_TABLE_PRIX structure

This structure contains the arrays for cash prices and cashless prices for each section of the VMC.

```
typedef struct
{
     unsigned int                uiPrixEspece;
     unsigned int                uiPrixCarte ;
     unsigned char               ucNumSelection ;
     unsigned char               ucValiditePrixEspece ;
     unsigned char               ucValiditePrixCarte ;
     unsigned char               ucRuf;
} T_UCMHOST_DA_TABLE_PRIXV3 ;
```

3.6.4 T_UCMHOST_DA_PARAM_MSGV3 structure

This structure contains messages displayed by the protocol DLL.

```
typedef struct
{
   unsigned char  tucIdleMsg                    [ 65 + 1 ] ; /*(1) Idle message */
   unsigned       char tucMsgDaNonInit          [ 65 + 1 ] ; /*(2) Dll not initialized */
   unsigned char  tucMsgCommunicationVmcKo      [ 65 + 1 ] ; /*(3) VMC communication Error*/
   unsigned char  tucMsgProductSelected         [ 65 + 1 ] ; /*(4) product selected from idle/
   unsigned char  tucMsgProductPriceNotDefined  [ 65 + 1 ] ; /*(5) Product doen't exist in price
                                                                  table */
}T_UCMHOST_DA_PARAM_MSGV3 ;
```

(1) : Idle message i.e. « Please\n Insert your card »
Some tags could be used to display the date.

"\dd"	:	to display the day
"\mm"	:	to display the month
"\yyyy" or "\yy"	:	to display the year
"\hh"	:	to display hours
"\ii"	:	to display minutes

for ex:
"Please\n Insert your card\n\\dd/\\mm/\\yyyy \\hh:\\ii"

(2) : Message displayed when the settings are not yet sent to the DLL protocol (see T_UCMHOST_DA_PARAMV2 structure) or when the field 'ucValidity' of T_UCMHOST_DA_PARAMV2 structure is set to 0 .

(3) : Message displayed when the communication with VMC doesn't work.

(4) : used with UCMHOST_VENDING_MODE_SELECTION_FIRST or UCMHOST_VENDING_MODE_PUSHBUTTON modes, This message is displayed by the DLL when a product is selected from idle state (this message could invite holder to present his card).
New -> With Executiv protocol, it is possible to display the price of the product selected. If the sub-string #P# is present in 'tucMsgProductSelected' string message, it is replaced by the price of the product.
The line displayed is centered.
If the line has more than 16 caractères, the first 16 caracters are displayed. i.e. "#P# EURO(S)\nPRESENT\nYOUR CARD".

(5) : this message is used in banking vending configuration when transaction is done before product selection.

3.6.5 MSG : UCMHOSTLIB_MSG_ASK_CHANGE_IDLE_MSG

This message is destined for UCMC. It allows to change Idle message.

```
typedef struct
{
    unsigned short        usWho;   /* 0 */
    unsigned short        usType;  /* UCMHOSTLIB_MSG_ASK_CHANGE_IDLE_MSG */
    int                   iStatus; /* 0 */
    unsigned int          uiNbApp; /* 0 */
    unsigned int          uiSize;  /* size of data pointed by pMessageRepos */

    union
    {
        unsigned char     *  pMessageRepos ;
    }u;

}T_UCMHOST ;
```

3.6.6 MSG : UCMHOSTLIB_MSG_ASK_REMOVE_CARD

This message allows to request the end of the session in progress.

```
typedef struct
{
    unsigned short        usWho;   /* 0 */
    unsigned short        usType;  /* UCMHOSTLIB_MSG_ASK_REMOVE_CARD */
    int                   iStatus; /* 0 */
    unsigned int          uiNbApp; /* 0 */
    unsigned int          uiSize;  /* 0 */

}T_UCMHOST ;
```

3.6.7 MSG : UCMHOSTLIB _MSG_NOT_AVAILABLE

This message allows to indicate that the protocol DLL is not available.

```
typedef struct
{
    unsigned short        usWho;   /* 0 */
    unsigned short        usType;  /* UCMHOSTLIB_MSG_NOT_AVAILABLE */
    int                   iStatus; /* 0 */
    unsigned int          uiNbApp; /* 0 */
    unsigned int          uiSize;  /* size of datas pointed by puiReason */

    union
    {
       unsigned int       * puiReason ;
    }u;

}T_UCMHOST ;
```

3.6.8 MSG : UCMHOSTLIB_MSG_ASK_DEBIT

This message allows to request e-purse debit.

```
typedef struct
{
   unsigned short         usWho;   /* 0 */
   unsigned short         usType;  /* UCMHOSTLIB_MSG_ASK_DEBIT */
   int                    iStatus; /* 0 */
   unsigned int           uiNbApp; /* 0 */
   unsigned int           uiSize;  /* size of data pointed by psDebit */
   union
   {
     T_UCMHOST_DEBIT      *  psDebit ;
   }u;
}T_UCMHOST ;
```

```
typedef struct
{
   unsigned long   ulAmount ;       /* (1) */
   S_MONEY         tCurrency;       /* (2) */
   unsigned char   ucTrsType;       /* 0 */
   unsigned char   ucTrsEntry;      /* 0 */
   unsigned char   ucTrsMode;       /* 0 */
   unsigned char   ucTrsSupport;    /* 0 */
   unsigned char   ucFunction;      /* UCMHOST_FCT_SOLV */
   unsigned char   ucMode;          /* 0 */
   unsigned char   ucClasse;        /* 0 */
   unsigned char   ucPrint;         /* 0 */
   unsigned char   ucDisplay;       /* 0 */
   unsigned short  usToWaitingCard; /* (3) */
   unsigned short  usToRemovedCard; /* 0 */
   unsigned char   ucAppliNum;      /* 0 */
   unsigned char   ucPowerOn;       /* 0 */
   union
   {
     T_UCMC_DA_ASK_DEBIT   tDaAskForDebit ;
   } u ;

} T_UCMHOST_DEBIT ;
```

```
typedef struct
{
   unsigned char          ucCmd  ;                    /* UCMCPAY_SOLV */
   unsigned char          ucSelectionNumber ;
   unsigned char          ucVendType ;                /* 1 if multi-vend enable else 0*/
   unsigned char          ucSelectionNotDefined ;     /* (4) */
   unsigned int           uiSelectionPrice ;
   unsigned char          tucCurrencyCode [ 3 ] ;
} T_UCMHOST_DA_ASK_DEBIT ;
```

(1)(2) : Amount and Currency is set in T_UCMHOST_DA_ASK_DEBIT struct and in the header.

(3) : Chip presentation Timeout (second) chip. The contactless application has to use this timeout to wait for chip presentation. This timeout is set from VMC settings (§3.6.14).

(4) :

If set to 1 :

- prices are defined in the vending machine.
- If protocole used is Executive ucSelectionNumber is inconsistent.
- If protocole used is MDB ucSelectionNumber corresponds to selection number of the selected product.

If set to 0xff :

- holding mode configuration.
- ucSelectionNumber corresponds to selection number of the selected product.

3.6.9 MSG : UCMHOSTLIB_MSG_CR_DISTRIBUTION

This message allows to indicate the return code of product distribution.

```
typedef struct
{
   unsigned short         usWho;   /* 0 */
   unsigned short         usType;  /* UCMHOSTLIB_MSG_CR_DISTRIBUTION */
   int                    iStatus; /* 0 */
   unsigned int           uiNbApp; /* 0 */
   unsigned int           uiSize;  /* size of data pointed by psDebit */
   union
   {
    T_UCMHOST_DEBIT       *  psDebit ;
   }u;
}T_UCMHOST ;

typedef struct
{
   unsigned long   ulAmount ;                    /* 0 */
   S_MONEY         tCurrency;                    /* 0 */
   unsigned char   ucTrsType;                    /* 0 */
   unsigned char   ucTrsEntry;                   /* 0 */
   unsigned char   ucTrsMode;                    /* 0 */
   unsigned char   ucTrsSupport;                 /* 0 */
   unsigned char   ucFunction;                   /* UCMHOST_FCT_ENREG */
   unsigned char   ucMode;                       /* 0 */
   unsigned char   ucClasse;                     /* 0 */
   unsigned char   ucPrint;                      /* 0 */
   unsigned char   ucDisplay;                    /* 0 */
   unsigned short  usToWaitingCard;              /* 0 */
   unsigned short  usToRemovedCard;              /* 0 */
   unsigned char   ucAppliNum;                   /* 0 */
   unsigned char   ucPowerOn;                    /* 0 */
   union
   {
    T_UCMHOST_DA_CR_DISTRIBUTION   tDaCrDistribution ;
   } u ;

} T_UCMHOST_DEBIT ;


typedef struct
{
   unsigned char     ucCmd ;                     /* UCMCPAY_RECORD */
   unsigned char     ucCrDistribution ;          /* 0 - Vend Succeeded  1-Vend Failed */
   unsigned char     ucSelectionNumber ;
} T_UCMHOST_DA_CR_DISTRIBUTION ;
```

3.6.10 MSG : UCMHOSTLIB_MSG_ASK_REVALUE

This message allows to request revalue e-purse.

```
typedef struct
{
   unsigned short      usWho;      /* 0 */
   unsigned short      usType;     /* UCMHOSTLIB_MSG_ASK_REVALUE */
   int                 iStatus;    /* message status */
   unsigned int        uiNbApp;    /* 0 */
   unsigned int        uiSize;     /* size of data pointed by psDebit */
   union
   {
    T_UCMHOST_DEBIT    *  psDebit ;
   }u;
}T_UCMHOST ;


typedef struct
{
   unsigned long   ulAmount ;      /* 0 */
   S_MONEY         tCurrency;      /* 0 */
   unsigned char   ucTrsType;      /* 0 */
   unsigned char   ucTrsEntry;     /* 0 */
   unsigned char   ucTrsMode;      /* 0 */
   unsigned char   ucTrsSupport;   /* 0 */
   unsigned char   ucFunction;     /* UCMHOST_FCT_REVALUE */
   unsigned char   ucMode; /* 0 */
   unsigned char   ucClasse;       /* 0 */
   unsigned char   ucPrint;        /* 0 */
   unsigned char   ucDisplay;      /* 0 */
   unsigned short  usToWaitingCard; /* 0 */
   unsigned short  usToRemovedCard; /* 0 */
   unsigned char   ucAppliNum;     /* 0 */
   unsigned char   ucPowerOn;      /* 0 */
   union
   {
    T_UCMHOST_DA_ASK_REVALUE       tDaAskRevalue ;
   } u ;

} T_UCMHOST_DEBIT ;


typedef struct
{
   unsigned char    ucCmd ; /* UCMC_REVALUE_EPURSE */
   unsigned char    tucCurrencyCode [ 3 ] ;
   unsigned int     uiRevalueAmount ;
} T_UCMHOST_DA_ASK_REVALUE ;
```

3.6.11 MSG : UCMHOSTLIB_MSG_REC_REVALUE

This message allows to record the e-purse revalue realised.

```
typedef struct
{
    unsigned short        usWho;       /* 0*/
    unsigned short        usType;      /* UCMHOSTLIB_MSG_ASK_REVALUE */
    int                   iStatus;     /* FCT_OK */
    unsigned int          uiNbApp;     /* 0 */
    unsigned int          uiSize;      /* size of data pointed by psDebit */
    union
    {
     T_UCMHOST_DEBIT      *  psDebit ;
    }u;
}T_UCMHOST ;
```

```
typedef struct
{
   unsigned long   ulAmount ;      /* 0 */
   S_MONEY         tCurrency;      /* 0 */
   unsigned char   ucTrsType;      /* 0 */
   unsigned char   ucTrsEntry;     /* 0 */
   unsigned char   ucTrsMode;      /* 0 */
   unsigned char   ucTrsSupport;   /* 0 */
   unsigned char   ucFunction;     /* UCMHOST_FCT_REVALUE */
   unsigned char   ucMode;         /* 0 */
   unsigned char   ucClasse;       /* 0 */
   unsigned char   ucPrint;        /* 0 */
   unsigned char   ucDisplay;      /* 0 */
   unsigned short  usToWaitingCard;/* 0 */
   unsigned short  usToRemovedCard;/* 0 */
   unsigned char   ucAppliNum;     /* 0 */
   unsigned char   ucPowerOn;      /* 0 */
   union
   {
    T_UCMHOST_DA_CR_REVALUE        tDaRecRevalue ;
   } u ;

} T_UCMHOST_DEBIT ;
```

```
typedef struct
{
   unsigned char      ucCmd ;        /* UCMC_RECORD_REVALUE_EPURSE */
   unsigned char      ucType ;
   unsigned char      tucRuf ;
   unsigned char      ucCrRevalue ; /* 0 - revalue Succeeded 1-revalue Failed */
} T_UCMHOST_DA_REC_EPURSE_REVALUE ;
```

3.6.12 MSG : UCMHOSTLIB_MSG_ASK_DISP_MSG_APPLI

This message allows to indicate to the component that messages to display have to be sent to the protocol DLL.

```
typedef struct
{
   unsigned short         usWho;               /* 0 */
   unsigned short         usType;              /* UCMHOSTLIB_MSG_ASK_DISP_MSG_APPLI */
   int                    iStatus;             /* FCT_OK */
   unsigned int           uiNbApp;             /* 0 */
   unsigned int           uiSize;              /* 0 */

}T_UCMHOST ;
```

3.6.13 MSG UCMHOSTLIB_MSG_PARAM_DA_MSG

This message allows to indicate the messages displayable by the protocol DLL.

```
typedef struct
{
   unsigned short         usWho;               /* 0 */
   unsigned short         usType;              /* UCMHOSTLIB_MSG_PARAM_DA_MSG */
   int                    iStatus;             /* FCT_OK */
   unsigned int           uiNbApp;             /* 0 */
   unsigned int           uiSize;              /* size of data pointed by u */
   union
   {
    T_UCMHOST_DA_PARAM_MSGV3     *  psParamDaMsg ;
   }u;

}T_UCMHOST ;


typedef struct

{
   unsigned char     tucIdleMsg                     [65+1] ;  /* Idle message */
   unsigned char     tucMsgDaNonInit                [65+1] ;  /* Dll not initialized */
   unsigned char     tucMsgCommunicationVmcKo       [65+1] ;  /* VMC communication error */
   unsigned char     tucMsgProductSelected          [65+1] ;  /*used in first selection mode*/
   unsigned char     tucMsgProductPriceNotDefined   [65+1] ;  /* T_UCMHOST_DA_PARAM_MSGV4 structure
                                                                 see § T_UCMHOST_DA_PARAM_MSGV4
                                                                 structure for more information*/

}T_UCMHOST_DA_PARAM_MSGV3 ;
```

3.6.14 MSG : UCMHOSTLIB_MSG_PARAM_DA

This message allows to send the parameters to the protocol DA.

```
typedef struct
{
   unsigned short    usWho;         /* 0 */
   unsigned short    usType;        /* UCMHOSTLIB_MSG_PARAM_DA */
   int               iStatus;       /* FCT_OK */
   unsigned int      uiNbApp;       /* 0 */
   unsigned int      uiSize;        /* size of data pointed by u */
   union
   {
    T_UCMHOST_DA_PARAMV4   *  psParamDa ;
   }u;
}T_UCMHOST ;
```

```
typedef struct
{
   unsigned char            ucValidity;
   unsigned char            tucTerminalNumber [ 10 + 1 ] ;
   unsigned char            ucVmcType [ 2 ] ; /*{0,0}*/
   unsigned short           usiTimeOutIfNoSelection ;
   unsigned short           usiTimeOutBuzzer ;
   unsigned short           usiBuzzerDuration ;
   unsigned char            ucDigitNumber ;
   unsigned char            tucCurrencyLabel [ 3 + 1 ] ;
   unsigned char            tucCurrencyCode [ 3 + 1 ] ;
   unsigned char            ucMultivendPossible;
   unsigned char            ucPriceHolding;
   unsigned char            ucRuf;
   unsigned int             usiScaleFactor;
   unsigned char            ucNbSelection;
   unsigned char            uctRuf2[ 3 ];
   T_UCMHOST_DA_TABLE_PRIX  tPriceTable [ 100 ] ;
   unsigned short           usTimeOutIfSelected ;
   unsigned char            ucVendingMode ;
   unsigned char            ucDisplayPrice;
}T_UCMHOST_DA_PARAMV4 ;
```

see § T_UCMC_DA_PARAMV4 structure for more information about this structure.

```
typedef struct
{
   unsigned int        uiPrixEspece;           /* price for sale by coins*/
   unsigned int        uiPrixCarte ;           /* price for sale by ICC */
   unsigned char       ucNumSelection ;        /* No of Selection  (>= 0 with MDB and >=1 with
                                                  EXE)*/
   unsigned char       ucValiditePrixEspece ;  /* 1 = the sale by coins is allowed */
   unsigned char       ucValiditePrixCarte ;   /* 1 = the sale by ICC is allowed */
   unsigned char       ucRuf;
}T_UCMHOST_DA_TABLE_PRIXV3 ;
```

3.6.15 MSG : UCMHOSTLIB_MSG_EPURSE_BALANCE

This message allows to send e-purse balance to the protocol DLL.

```
typedef struct
{
   unsigned short         usWho;              /* 0 */
   unsigned short         usType;             /* UCMHOSTLIB_MSG_EPURSE_BALANCE */
   int                    iStatus;            /* FCT_OK */
   unsigned int           uiNbApp;            /* 0 */
   unsigned int           uiSize;             /* size of data pointed by u */
   union
   {
    T_UCMHOST_DA_EPURSE_BALANCEV3      tEpurseBalance ;
   }u;
}T_UCMHOST ;
```

```
typedef struct
{
    unsigned long     ulEpurseBalance ;
    unsigned char     ucCr ;                  /* 0 – Epurse OK     1- Epurse KO remove card */
    unsigned char     tucCurrencyCode [ 3 ] ;
    unsigned char     tucLanguageCode [ 3 ] ;
    unsigned char     ucAllowRevalue ;        /* 0 revalue disabled  1 revalue enabled*/
    unsigned char     ucAllowRefund;          /* 0 disabled 1 enabled*/
    unsigned char     ucAllowDisplayBalance ;
    unsigned char     ucAllowMultiVend ;      /* 0 multivend disabled 1 multivend supported*/
    unsigned char     ucRuf ;
    unsigned long     ulRevalueLimitBalance ; /*set only if ucAllowRevalue!=0*/
    unsigned long     ulRevalueLimitAmount ;  /*set only if ucAllowRevalue!=0*/
} T_UCMHOST_DA_EPURSE_BALANCEV3 ;
```

Note:

o If ucCr is not null, the DLL goes to card removal state (buzzer and red Led blinking). The application has to send UCMHOSTLIB_MSG_END when the card is removed

o UcAllowRevalue is set to 1 if the application is able to credit card (from coin for example)

o UcAllowRefund is set to 1 if the application is able to refund sale when the sale is failed

o UcDisplayBalance is set to 1 to display the credit on VMC display (only with MDB and function supported by VMC)

o UcAllowMultivend is set to 0 to disable multi vend. The multivend mode is set by the setting application. If the debit application doesn't want to support multivend, UcAllowMultivend is set to 0. If set to 1, the multivend is enable if setting application enable it.

o When the revalue is enabled, ulRevalueLimitBalance and ulRevalueLimitAmount inform to DLL the additionnal conditions to allow the revalue.
UlRevalueLimitBalance is the max balance of the Epurse (5000 for 50EUR for ex)
UlRevalueLimitAmount is the max amount that can be credited on the Epurse (200 for coin 2€ for ex).However, when it receives a revalue request, the application has to check again the conditions to allow or refuse the request.

3.6.16 MSG : UCMHOSTLIB_MSG_ANSW_DEBIT

This message gives the return code of the e-purse debit.

```
typedef struct
{
   unsigned short         usWho;                 /* 0 */
   unsigned short         usType;                /* UCMHOSTLIB_MSG_ANSW_DEBIT */
   int                    iStatus;               /* FCT_OK */
   unsigned int           uiNbApp;               /* 0 */
   unsigned int           uiSize;                /* size of data pointed by u */
   union
   {
    T_UCMHOST_R_DEBIT    *  psDebit ;
   }u;
}T_UCMHOST ;


typedef struct
{
   unsigned char          ucCr;                  /* 0 */
   unsigned char          ucDiag ;               /* error code if error */
   unsigned char          ucUCMDiag ;            /* Init by UCMC :
                                                    0 = OK,
                                                    1 = Service not called,
                                                    2 = Called service returns an error,
                                                    3 = Application number */
   unsigned char          ucPrinter;             /* 0 */
   unsigned char          ucDisplay;             /* 0 */
   unsigned char          ucCardInside;          /* 1 = Card inside during transaction */
   unsigned char          ucMode;  /* 0 */
   unsigned char          ucFunction ;           /* UCMHOST_FCT_SOLV */
   unsigned char          ucTypeCardStruct ;     /* 0 */
   unsigned char          ucSupport ;            /* 0 */
   unsigned short         usAppName ;            /* segment number */
   T_AFFNOM               tAppLibelle;           /* application name */
   MONTANT                ulAmount;              /* transaction amount */
   S_MONEY                tCurrency;             /* currency of the transaction */
   unsigned char          ucCardHolderLanguage;  /* 0 */

   union
   {
    unsigned char ucBuf[ 20 ] ;
   } uRuf;

   union
   {
      T_UCMHOST_CARD      sCard ;
      T_UCMHOST_R_DEBIT_DA sRDebitDa ;
      unsigned char       ucBuf [ UCMHOST_MAX_SIZE_CARD_APPLI_INFO +
                                  UCMHOST_MAX_SIZE_CARD_INFO +
                                  UCMHOST_MAX_SIZE_CARD_ACCEPT_INFO ] ;
   }u;
} T_UCMHOST_R_DEBIT ;

typedef struct
{
   MONTANT                ulEpurseBalance ;
} T_UCMHOST_R_DEBIT_DA ;
```

3.6.17 MSG : UCMHOSTLIB_MSG_ANSW_REVALUE

This message allows to transmit the return code of e-purse revalue.

```
typedef struct
{
   unsigned short          usWho;              /* 0 */
   unsigned short          usType;             /* UCMHOSTLIB_MSG_ANSW_REVALUE */
   int                     iStatus;            /* FCT_OK */
   unsigned int            uiNbApp;            /* 0 */
   unsigned int            uiSize;             /* size of data pointed by u */
   union
   {
    T_UCMHOST_DA_CR_EPURSE_REVALUEV3    tEpurseRevalue ;
   }u;
}T_UCMHOST ;
```

```
typedef struct
{
  unsigned char            ucCrRevalue ;
  unsigned char            tucRuf [ 3 ] ;
  unsigned long            ulRevalueAmount ;
  unsigned long            ulEpurseBalance ;
} T_UCMHOST_DA_CR_EPURSE_REVALUEV3 ;
```

3.6.18 UCMHOSTLIB_MSG_END

This message is send to DLL to terminate or cancel the session.

```
typedef struct
{
   unsigned short     usWho;                  /* 0 */
   unsigned short     usType;                 /* UCMHOSTLIB_MSG_END */
   int                iStatus;                /* FCT_OK */
   unsigned int       uiNbApp;                /* 0 */
   unsigned int       uiSize;                 /* 0 */

}T_UCMHOST ;
```