Référence / Reference : SMO/SPE-309 Révision / *Revision* : B

Titre /*Title* : INTERFACE BETWEEN A BANKING APPLICATION AND A BANKING DLL PROTOCOL

Programme / *Subject* : **- UCM TELIUM -**

Approbation de la révision / *Revision Approval* : B		
	Nom ***Name***	**Fonction** ***Function***
Etabli par : *Written by:*	C. PLESSIS	UCMC Technical project leader
Vérifié ou Approuvé par : *Checked or approved by:*	Y. MORENO	Banking Technical project leader
Autorisé par : *Authorized by:*	A. SOUBIRANE	CAD 30 UCM Product manager

Révision *Issue*	**Date de validité / d'application** *Validity/application date*	**Nb de pages** *Nb of pages*	**Nb de pages annexes** *Nb of appendices*	**Objet et description de la modification** *Object and description of modification*
	12/01/2008	35		First version
A	01/07/2008	35		English corrections
B	09/10/2009	36		Adding Host simulation by Application

SUMMARY

1. INTRODUCTION

1.1 DOCUMENT PURPOSE

The purpose of this document is to describe the banking payment transaction flow and how to develop a banking application or a DLL banking protocol.

1.2 INPUT DATA

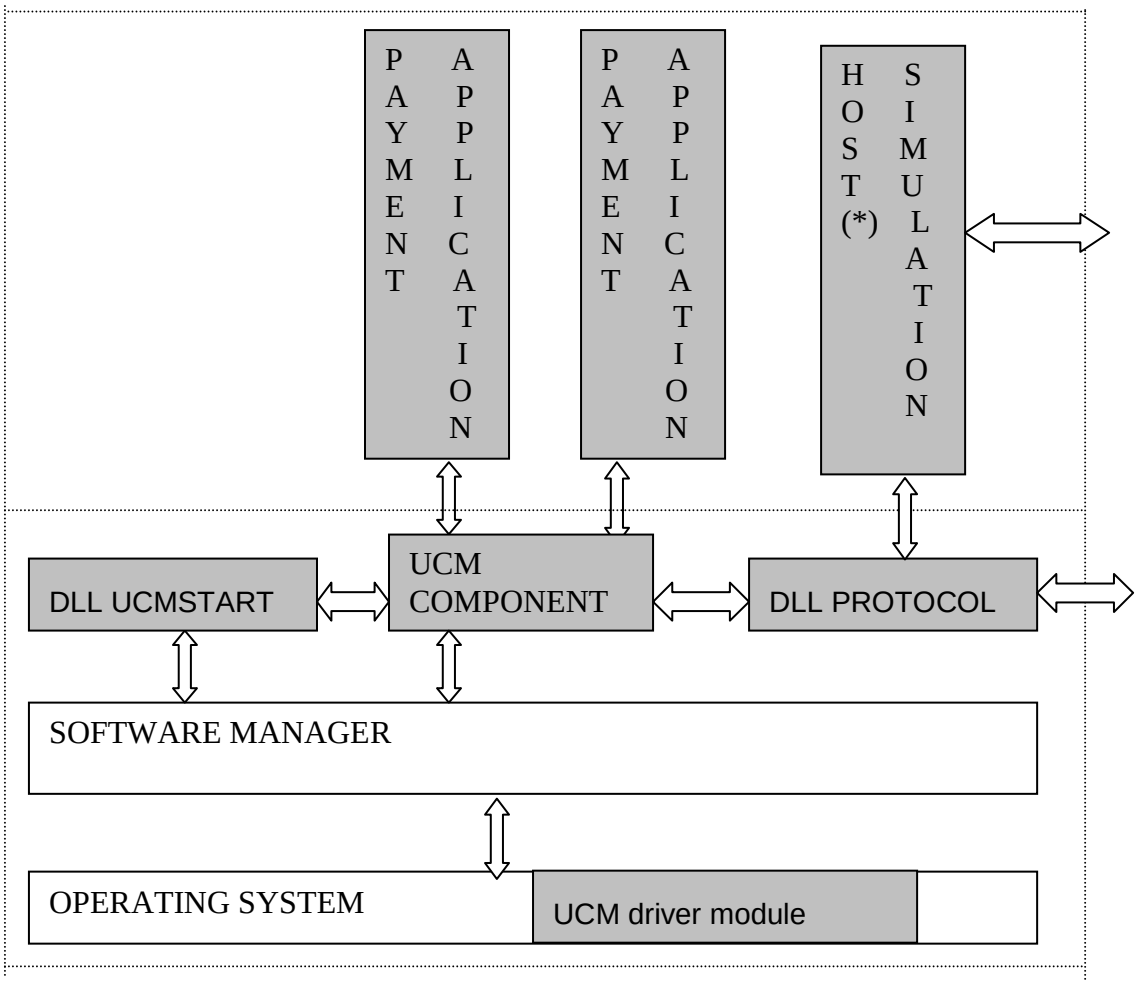
SMO/SFO-00069 : UCM component reference manual.

1.3 TERMINOLOGY

IAC	Interface Application Code
Reader	Chip card reader terminal used for payment.
UCM	Universal Communication Module.
UCMC	UCM Component

2. INTRODUCTION TO THE SOFTWARE

2.1 BREAKDOWN



Shaded area: UCM platform-specific software.

(*) : option by UCMC parameters

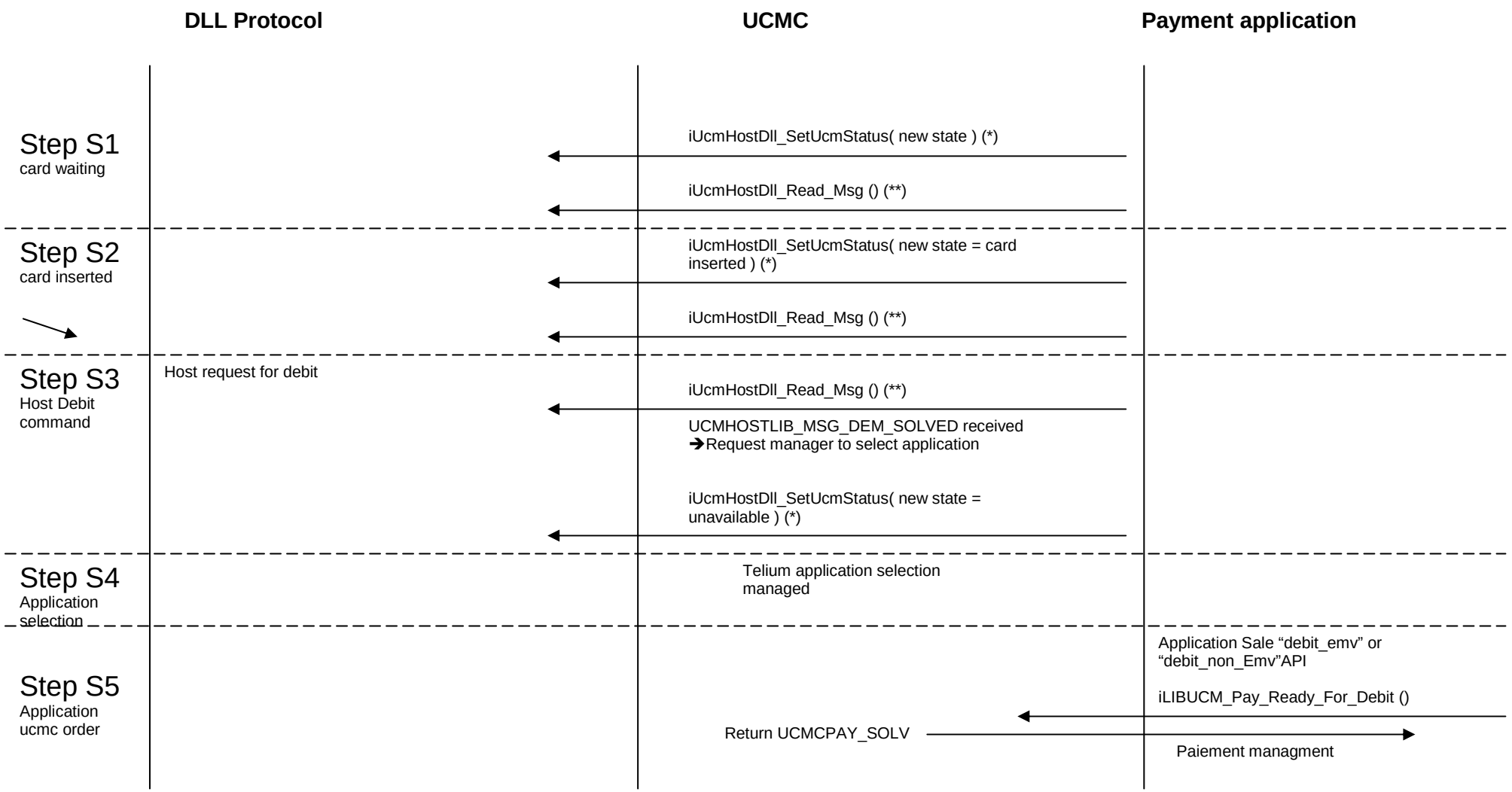
2.2 PRINCIPLE

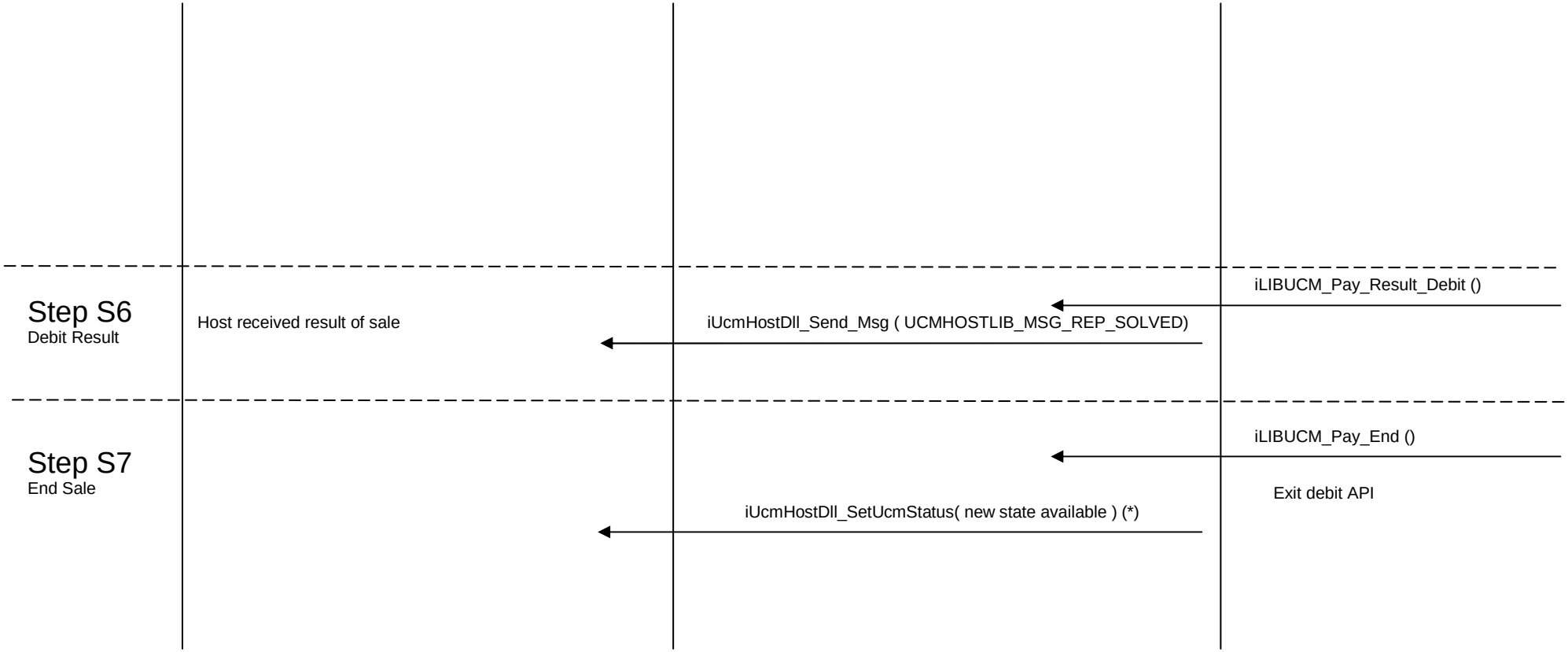
Payment applications use UCM component to access device and to process transaction.

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 debit, ask record…). Once the treatment finished, the application is put on standby and wait for a card event.

3. BANKING TRANSACTION FLOW

3.1.1 Debit cycle

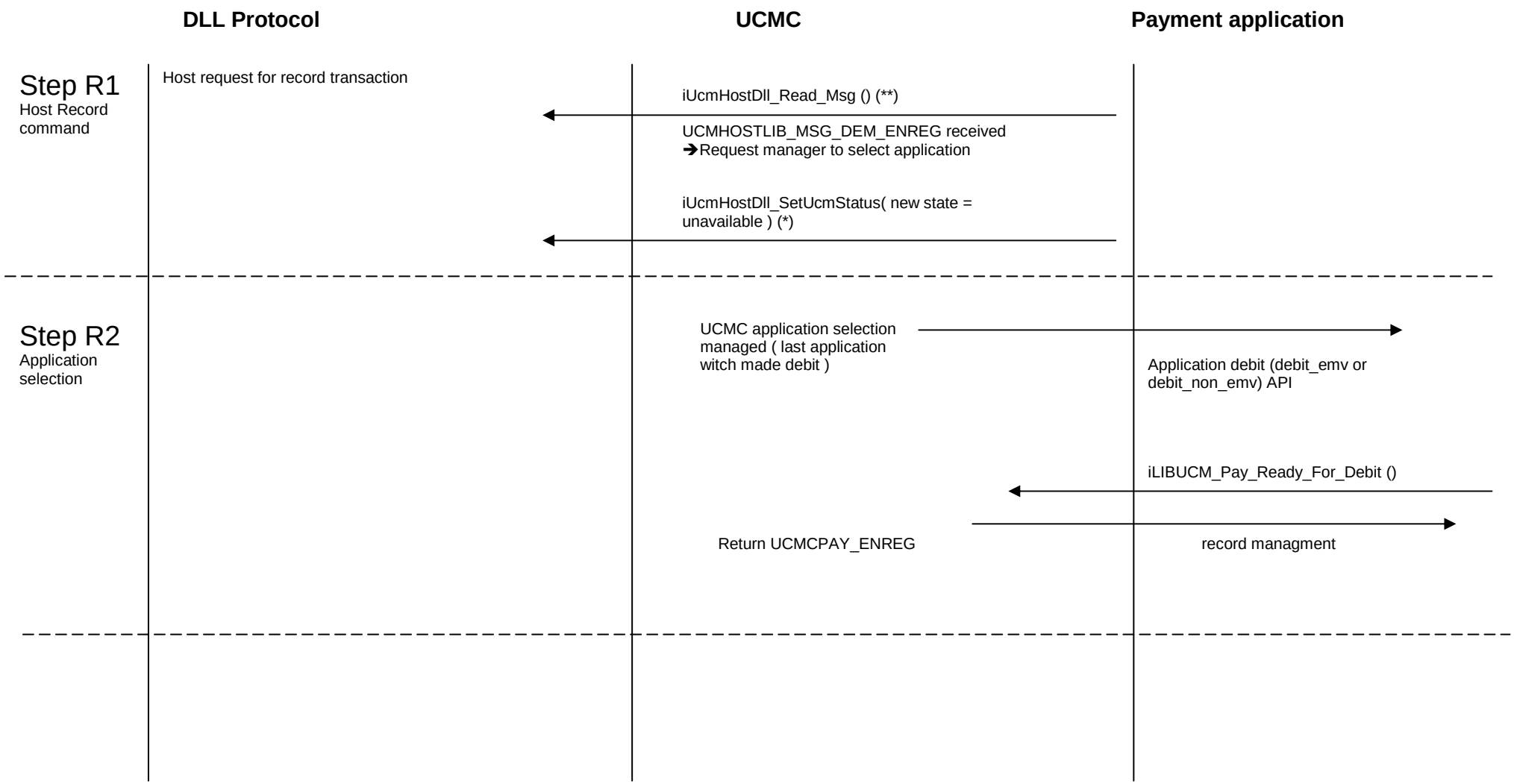


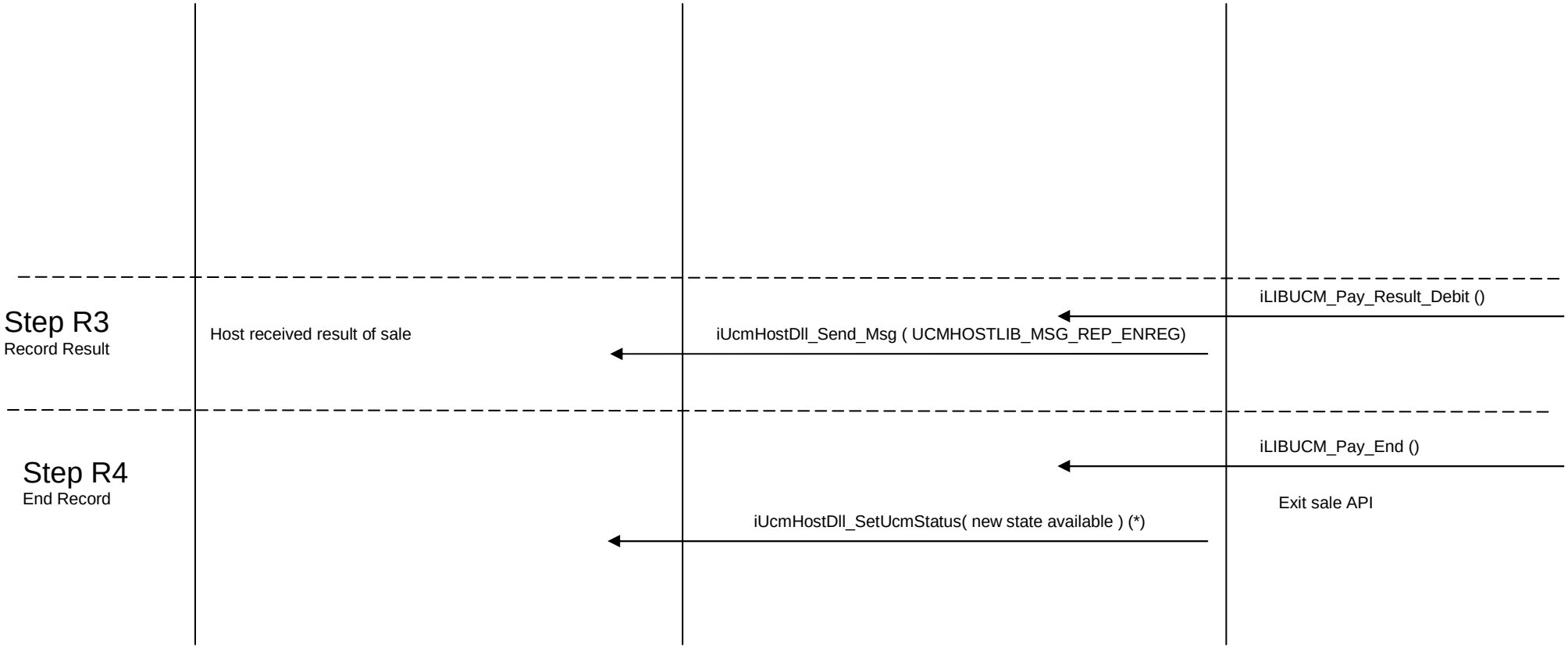


(*) State message : The state message is posted at each state change.

(**) Read message : UCMC read message is posted at each change.

3.1.2 Record cycle





(*) State message : The state message is posted at each state change.

(**) Read message : UCMC read message is posted at each change.

4. INTERFACE BETWEEN PAIEMENTS APPLICATIONS AND UCM COMPONENT

4.1 FUNCTIONS OF THE UCM COMPONENT LIBRARY

All API are Telium compatible.

The UCM Component places at the disposal of applications a library which offers services for the management of the peripherals connected to the UCM in order to realise payments transactions on a banking machine. Only aspect of payments transaction flow is described.

4.1.1 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 process to realize on the card (debit or record)

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.

UCMCPAY_SOLV, debit request.

UCMCPAY_RECORD, record transaction request.

UCMCPAY_SOLV_LOC, debit request for application managed record number.

UCMCPAY_RECORD_LOC, record transaction for application managed record number.

UCMCPAY_CARD_INFO, returns information of card introduction.

UCMERR_IAC_FCT_RETURN if payment not authorized.

4.1.2 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 or record 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 operation is correct.
negative value in case of error.

4.1.3 iLIBUCM_Pay_End

syntax : int iLIBUCM_Pay_End (int iSize_p, void * ps_p)

description : inform the UCMC of the exit of application API payment.

Parameters : iSize_p = 0.
ps_p = NULL.

Returns : the returned value is FCT_OK.

4.1.4 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 host 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.

In this configuration only request consolidation of transaction to the host may be used.

5. INTERFACE BETWEEN A PROTOCOL DLL AND UCM COMPONENT

5.1 PRINCIPLE

The protocol DLL must be compatible with the HOST UCM library.

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

The DLL is always called to dialog. The DLL pushes his messages into the FIFO which is periodically read by UCMC (every 20ms).

5.2 PROTOCOLE DLL Entry functions

5.2.1 LIBRARY API

The DLL must implement only 4 API :

- int iUcmHostLib_Open(char *);
- void UcmHostLib_Close(void);
- int iUcmHostLib_GetInfo(object_info_t *pinfos_p) ;
- int iUcmHostLib_Cmd(unsigned short, int, void*, int, void*) ;

These API are not described. These API are provided by UCMHOST.LIB library and only called by UCMC.

5.2.2 UcmHostDll_GetVersion()

syntax : void UcmHostDll_GetVersion(unsigned char *version)

description : Give version.

parameters : Version; Max=5 caracteres + null. Example : « 01.02 »

returns : None

This function is reserved for future use. UCMC calls directly the system to have real DLL version.

5.2.3 UcmHostDll_Cmd ()

syntax : void UcmHostDll_Cmd(int *piRet_p, unsigned short usCmd_p, int iLg1_p, void* vpPar1_p, int iLg2_p, void* vpPar2_p)

description : Send or receive messages or commands.

parameters : piRet_p: Result of commands as FTC_OK, UCMHOSTLIB_ERR_CMD_NOT_AUTHORIZED ….

usCmd_p: Commands as UCMHOST_INIT ….

iLg1_p : length of data1

vpPar1_p : pointers of data1; Depends of commands

iLg2_p : length of data2

vpPar2_p : pointers of data2; Depends of commands

returns : None

5.3 COMMANDS SENT TO PROTOCOL DLL BY UCMC

5.3.1 Introduction

These commands are coming from entry point UcmHostDll_Cmd().

Some are simple (UCMHOST_INIT …) and others need parameters.

5.3.2 UCMHOST_CLEAR

Clear Host buffers of reception and emission. Return FCT_OK

5.3.3 UCMHOST_CLOSE

Close communication with host. Free COM used.

5.3.4 UCMHOST_INIT

Initialize the DLL with information given by vpPar1_p of UcmHostDll_Cmd() function.

vpPar1_p points to T_UCM_DEVICE structure.

5.3.5 UCMHOST_SEND_MSG

vpPar1_p points to T_UCMHOST structure.

Information depends on message posted : usType from T_UCMHOST structure:

- UCMHOSTLIB_MSG_REP_STATUS: Status response. Not Used. The UCM periodically sends new status. If the host need a status, the protocol DLL must use information given by command UCMHOST_UCM_STATUS.
- UCMHOSTLIB_MSG_REP_ISO2: Track card reading. response from message UCMHOSTLIB_MSG_DEM_ISO2. Use T_UCMHOST_READ_TRACK structure.
- UCMHOSTLIB_MSG_REP_FCTAPP: Response from application function. UCMC sends response to several applications. Result from message UCMHOSTLIB_MSG_DEM_FCTAPP. Use T_UCMHOST_FCTAPP structure. Protocol DLL must memorise function requested to know which structure used.
    - UCMHOST_FCTAPP_CONSULT: use T_UCMHOST_APP_CONSULT.
    - UCMHOST_FCTAPP_TLC_START: use T_UCMHOST_APP_TLC_START.

- UCMHOST_FCTAPP_TLC_STATUS : use T_UCMHOST_APP_TLC_STATE
- UCMHOST_FCTAPP_TLP_STATUS : use T_UCMHOST_APP_TLP_STATE
- UCMHOST_FCTAPP_NB_RECORD_LOC: Specific. Not used.
- UCMHOST_FCTAPP_INFO_RECORD_LOC: Specific. Not used.
- UCMHOST_FCTAPP_LIST_RECORD_LOC: Specific. Not used.

- UCMHOSTLIB_MSG_REP_MTNC: Response to maintenance request. This mode is not authorized. Use T_UCMHOST_R_MTNC structure.
- UCMHOSTLIB_MSG_REP_SOLVED: Debit result. Response from UCMHOSTLIB_MSG_DEM_SOLVED message. Use T_UCMHOST_R_DEBIT structure.
- UCMHOSTLIB_MSG_REP_SOLVED_LOC: Not used. Specific.
- UCMHOSTLIB_MSG_REP_ENREG: Record result. Response from UCMHOSTLIB_MSG_DEM_ENREG message. Use T_UCMHOST_R_DEBIT structure.
- UCMHOSTLIB_MSG_REP_ENREG_LOC: Not used. Specific.
- UCMHOSTLIB_MSG_REP_CARD_INFO: Smart card Information. Response from UCMHOSTLIB_MSG_DEM_ CARD_INFO message. Use T_UCMHOST_R_DEBIT structure.
- UCMHOSTLIB_MSG_DEM_CONSO: Application requests transaction consolidation to the Host. Use T_UCMHOST_CONSO.
- UCMHOSTLIB_MSG_REP_CANCEL: Cancel action if possible. Application must read Host message to know this Host request. Response from UCMHOSTLIB_MSG_DEM_CANCEL message. Use first byte of u.pucData. Value are: UCMHOST_CANCEL_OK, UCMHOST_CANCEL_IMPOSSIBLE, UCMHOST_CANCEL_KO_MTNC or UCMHOST_CANCEL_KO_REFUSED (by application or UCMC):
- UCMHOSTLIB_MSG_REP_RESTART: Restart CAD30. Response from UCMHOSTLIB_MSG_DEM_RESTART message. Use T_UCMHOST_R_RESTART. IStatus = 0 .
- UCMHOSTLIB_MSG_DEM_CONNECT: Not Used. Use direct command UCMHOST_MODEM_CONNECT.
- UCMHOSTLIB_MSG_DEM_DISCONNECT: Not used. Use direct command UCMHOST_MODEM_DISCONNECT.
- UCMHOSTLIB_MSG_REP_CHGT_DATE: Change date. UCMC sends message to manager. Response from UCMHOSTLIB_MSG_DEM_ CHGT_DATE message. Use T_UCMHOST_NEW_DATE.

5.3.6 UCMHOST_READ_MSG

The UCMC reads information coming from Protocol DLL.

vpPar1_p points to T_UCMHOST structure.

Information depends on message posted : usType from T_UCMHOST structure:

- UCMHOSTLIB_MSG_DEM_SOLVED: Host debit request. Use T_UCMHOST_DEBIT structure. If there is no card, UCMC waits for card introduction.
- UCMHOSTLIB_MSG_DEM_ENREG: Not used. Host record request. Use T_UCMHOST_DEBIT structure.
- UCMHOSTLIB_MSG_DEM_CARD_INFO: Request Smart card information. Use T_UCMHOST_R_DEBIT structure. Used by specific Host and application.
- UCMHOSTLIB_MSG_DEM_STATUS: Request for status. Do not use this command because UCMC already sends status at each change.
- UCMHOSTLIB_MSG_DEM_VERSION: Ruf.
- UCMHOSTLIB_MSG_NO: Ruf.
- UCMHOSTLIB_MSG_DEM_RESTART: Request for restart. UCMC sends response UCMHOSTLIB_MSG_REP_RESTART. The protocol DLL must close his COM because UCMC attempts the downloading mode “compatible software LLT” during u.pucData bytes (seconds) before restarting.
- UCMHOSTLIB_MSG_REP_CONSO: Response from application request transaction consolidation. Use T_UCMHOST_CONSO for response.
- UCMHOSTLIB_MSG_DEM_CANCEL: Cancel request. Not always possible. Depends if application reads Host message.
- UCMHOSTLIB_MSG_DEM_MTNC: Not authorized.
- UCMHOSTLIB_MSG_DEM_FCTAPP:Request for application functions. UCMC sends request to all applications. Use T_UCMHOST_FCTAPP structure. Protocol DLL must memorise function requested to know which structure to use during response.
    - UCMHOST_FCTAPP_CONSULT: use T_UCMHOST_APP_CONSULT.
    - UCMHOST_FCTAPP_TLC_START: use T_UCMHOST_APP_TLC_START.
    - UCMHOST_FCTAPP_TLC_STATUS : use T_UCMHOST_APP_TLC_STATE
    - UCMHOST_FCTAPP_TLP_STATUS : use T_UCMHOST_APP_TLP_STATE
    - UCMHOST_FCTAPP_NB_RECORD_LOC: Specific. Not used.
    - UCMHOST_FCTAPP_INFO_RECORD_LOC: Specific. Not used.
    - UCMHOST_FCTAPP_LIST_RECORD_LOC: Specific. Not used.
- UCMHOSTLIB_MSG_DEM_CHGT_DATE: Change date. UCMC sends message to manager. Use T_UCMHOST_NEW_DATE.
- UCMHOSTLIB_MSG_DEM_ISO2: Read track 2.
- UCMHOSTLIB_MSG_BUZZER: Buzzer command. Not used in banking.
- UCMHOSTLIB_MSG_DISPLAY: RUF.

- UCMHOSTLIB_MSG_REP_CONNECT: Connection to banking host response. Response for manager. Use T_UCMHOST_R_CONNECT structure.
- UCMHOSTLIB_MSG_NETWORK_READ: Read Data from banking host to manager. No structure. Data provide directly from banking Host.
- UCMHOSTLIB_MSG_DEM_SYS: Ruf.
- UCMHOSTLIB_MSG_DEM_CONNECT: Ruf.
- UCMHOSTLIB_MSG_DEM_DISCONNECT: Ruf.
- UCMHOSTLIB_MSG_DEM_NETWORK_WRITE: Ruf.
- UCMHOSTLIB_MSG_DEM_NETWORK_STATUS: Ruf.
- UCMHOSTLIB_MSG_DEM_PRINTER: Ruf.
- UCMHOSTLIB_MSG_DEM_ICC: Ruf.
- UCMHOSTLIB_MSG_DEM_PPAD: Ruf.
- UCMHOSTLIB_MSG_LED: Not used for banking. For vending.
- UCMHOSTLIB_MSG_ALL_LEDS: Not used for banking. For vending.
- UCMHOSTLIB_MSG_ASK_DEBIT: Not used for banking. For vending.
- UCMHOSTLIB_MSG_CR_DISTRIBUTION : Not used for banking. For vending.
- UCMHOSTLIB_MSG_ASK_REMOVE_CARD : Not used for banking. For vending.
- UCMHOSTLIB_MSG_ASK_REVALUE : Not used for banking. For vending.
- UCMHOSTLIB_MSG_REC_REVALUE: Not used for banking. For vending.
- UCMHOSTLIB_MSG_ASK_CHANGE_IDLE_MSG: Not used for banking. For vending.

5.3.7 UCMHOST_DLL_STATUS

UCMC requests Protocol DLL status. vpPar1_p points to T_UCMHOST_MPA_STATUS structure. This structure must be filled by Protocol DLL.

5.3.8 UCMHOST_UCM_STATUS

UCMC gives status at each change. vpPar1_p points to T_UCMHOST_STATUS_UCM structure.

5.3.9 UCMHOST_STOP_REC

Stop protocol reception. Com is not closed.

5.3.10 UCMHOST_START_REC

Restart protocol reception.

5.3.11 UCMHOST_DISPLAY_MSG

Display message on Host display. Not available on CAD30UPT series.

Message are compatible 2 lines of 16 characters each.

5.3.12 UCMHOST_PRINT_MSG

Print message on Host printer.

Message are ascii compatible.

5.3.13 UCMHOST_LED_MSG

Message for Led managed by Host.

Request coming from applications. See T_UCMC_IAC_LED. Not Used.

5.3.14 UCMHOST_BUZZER_MSG

Message for buzzer managed by Host.

Request coming from specific applications. See T_UCMC_IAC_BUZ. Not Used.

5.3.15 UCMHOST_ICC_MSG

Message for card reader managed by Host.

Not Used.

5.3.16 UCMHOST_PINPAD_MSG

Message for pinpad managed by Host.

Not Used.

5.3.17 UCMHOST_MODEM_CONNECT

Modem is managed by Host.

This command is sent by manager. Use T_UCMHOST_D_CONNECT structure.

5.3.18 UCMHOST_MODEM_DISCONNECT

Modem is managed by Host.

This command is sent by manager. No structure.

5.3.19 UCMHOST_MODEM_WRITE

Modem is managed by Host.

Data are directly sent.

5.3.20 UCMHOST_MODEM_READ

Modem is managed by Host.

Data are directly read.

5.3.21 UCMHOST_MODEM_STATUS

Modem is managed by Host.

Status of modem.

5.3.22 COMMANDS NOT POSTED by UCMC

The DLL must return UCMHOSTLIB_ERR_CMD_NOT_AUTHORIZED for the following commands:

- UCMHOST_MASK:
- UCMHOST_WAIT_MSG:
- UCMHOST_SEND:
- UCMHOST_READ:
- UCMHOST_START:
- UCMHOST_STOP:
- UCMHOST_SUSPEND:
- UCMHOST_TEST:

5.4 MESSAGES POSTED BY PROTOCOL DLL TO UCMC

The protocol DLL never calls UCMC. The protocol pushes messages in FIFO which is periodically (20ms) read by UCMC.

See chapter “COMMANDS SEND TO PROTOCOL DLL BY UCMC” command UCMHOST_READ_MSG.

6. STRUCTURE USED

6.1 INTRODUCTION

All structures are implemented in file ucmhostdll.h

Each message exchanged between DLL Protocol and UCMC uses T_UCMHOST structure. The structure fields are function of the type of message (union). Below is described the T_UCMHOST messages according with the message type.

6.1.1 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_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_DA_PARAM                     *  pParamDa ;
      T_UCMHOST_DA_PARAM_MSG                 *  pParamDaMsg ;
      T_UCMHOST_DA_EPURSE_BALANCE            *  pEPurseBal ;
      T_UCMHOST_DA_CR_EPURSE_REVALUE         *  pCrRevalue ;
      T_UCMHOST_DA_CR_REC_EPURSE_REVALUE     *  pCrRecRevalue ;
      unsigned int                           *  puiReason ;
  }u;
}T_UCMHOST ;
```

6.2 T_UCMHOST_DEBIT STRUCTURE : DEBIT OR RECORD REQUEST

This structure is filled by DLL protocol and UCMC.

It's read in application using iLIBUCM_Pay_Ready_For_Debit()

```
typedef struct
{
   unsigned long   ulAmount ;      /* Fill by DLL protocol */
   S_MONEY         tCurrency;      /* Fill by DLL protocol and UCMC */
   unsigned char   ucTrsType;      /* Fill by DLL protocol: Used by UCMC */
   unsigned char   ucTrsEntry;     /* Fill by DLL protocol: Used by UCMC */
   unsigned char   ucTrsMode;      /* Fill by DLL protocol: Used by UCMC */
   unsigned char   ucTrsSupport;   /* Fill by DLL protocol: Used by UCMC */
   unsigned char   ucFunction;     /* UCMHOST_FCT_SOLV, UCMHOST_FCT_ENREG */
   unsigned char   ucMode;         /* Fill by DLL protocol: Used by Appli */
   unsigned char   ucClasse;       /* Fill by DLL protocol: Used by Appli */
   unsigned char   ucPrint;        /* Fill by DLL protocol: Used by Appli */
   unsigned char   ucDisplay;      /* Fill by DLL protocol: Used by Appli */
   unsigned short  usToWaitingCard;/* Fill by DLL protocol */
   unsigned short  usToRemovedCard;/* Fill by DLL protocol */
   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 ;      /* Vending */
      T_UCMC_DA_CR_DISTRIBUTION    tDaCrDistribution ;   /* Vending */
      T_UCMC_DA_ASK_REVALUE        tDaAskForRevalue ;    /* Vending */
      T_UCMC_DA_REC_EPURSE_REVALUE tDARecRevalue ;       /* Vending */
      T_UCMHOST_SOLV_COMP_LOC      tSolvLoc ;            /* Reserve */
   } u ;

} T_UCMHOST_DEBIT ;
```

This structure may use sub-structures T_UCMHOST_SOLV_COMP_LOC only for Host and application using local record number.

Definition of each field:

- ulAmount : Amount filled by Protocol DLL (example 1234 for 12,34 Euros)
- tCurrency.code: filled by DLL Protocol (example 978 for Euro).
- tCurrency.posdec : filled by DLL Protocol (example 2)
- tCurrency.nom : filled by UCMC using currency of manager (EUR for 978).
- UcTrsType : Type of transaction see TRANSACTION_TYPE (appel.h). Used by UCMC to fill transin.transaction. Filled by DLL Protocol to DEBIT_TR.
- UcTrsEntry : Type of entry see ENTRY_TYPE (appel.h). Used by UCMC to fill transin.entry. Filled by DLL Protocol to NO_ENTRY.
- UcTrsMode : Mode of transaction see PAYMENT_MODE (appel.h). Used by UCMC to fill transin.payment. Filled by DLL Protocol to UNKNOWN_PAYMENT.

- UcTrsSupport : Type of support see SUPPORT_TYPE (appel.h). Used by UCMC to fill transin.support but update by manager. Filled by DLL Protocol to UNKNOWN_SUPPORT.
- UcFunction : Filled by Protocol DLL. UCMHOST_FCT_SOLV for debit or UCMHOST_FCT_ENREG for record. UCMHOST_FCT_SOLV_LOC, UCMHOST_FCT_ENREG_LOC and UCMHOST_FCT_CARD_INFO are used only by applications using local record number.
- UcMode : Filled by Protocol DLL to UCMHOST_MODE_TEST or UCMHOST_MODE_REEL. Used only by application.
- UcClasse : Filled by Protocol DLL with '2' for class '2.1'. Used only by application.
- UcPrint : Filled by Protocol DLL with default value 0xFF. Then updates by UCMC with UCMC parameters 061402 if it's the default value; Value 0x01 means application managed banking printer (located in host or connected).
- UcDisplay : Filled by Protocol DLL with default value 0xFF. Then updates by UCMC with UCMC parameters 061002 if it's the default value; Not used.
- UsToWaitingCard : Filled by Protocol DLL with default value 60. If the card is not inside, the UCMC waits this card for 60 seconds. If the card is not present after 60s a negative response is sent to Protocol DLL.
- UsToRemovedCard : Filled by Protocol DLL with a value in second. Used by application to wait the removed card at the end of debit entry point.
- UcAppliNum : Filled by Protocol DLL with default value 0. Used by UCMC to fill transin.cardapplinum but updated by manager.
- UcPowerOn : Filled by Protocol DLL with value 1 for debit or 0 for record. Used by UCMC to fill transin.power_on_result but update by manager.

6.2.1 T_UCMHOST_R_DEBIT STRUCTURE : DEBIT OR RECORD response

```
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;  /* 0 if printer OK */
      unsigned char     ucDisplay;  /* 0 if display OK */
      unsigned char     ucCardInside;     /* 1 = Card inside during transaction */
      unsigned char     ucMode;     /* NA */
      unsigned char     ucFunction ;      /* UCMHOST_FCT_SOLV, UCMHOST_FCT_ENREG */
      unsigned char     ucTypeCardStruct ;      /* UCMHOST_CARD_EMV ... */
      unsigned char     ucSupport ; /* see SUPPORT_TYPE */
      unsigned short    usAppName ; /* application segment number*/
      T_AFFNOM    tAppLibelle;      /* application name i.e PME SELECTA */
      MONTANT           ulAmount;   /* transaction amount */
      S_MONEY           tCurrency;  /* currency of the transaction */
      unsigned char     ucCardHolderLanguage;   /* NA */

      union
      {
            unsigned char ucBuf[ 20 ] ; /* Not used */
            T_UCMHOST_SOLV_COMP_LOC sLoc;

            } 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 ;
```

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.

Definition of each field:

- ucCr : UCMHOST_CR_OK or other value if error.: UCMHOST_CR_BUSY, UCMHOST_CR_MTNC …. Some values can be set by applications or UCMC if applications not reached. See error chapter.
- UcDiag : UCMHOST_CR_OK if OK. Some values can be set by applications or UCMC if applications not reached. See error chapter.
- UcUCMDiag : Set to 0 by applications. Reserve for UCMC if error.
- UcPrinter : Status of printer. Set by applications. Set by UCMC if applications not reached: 0 = OK. 1 = KO.
- UcDisplay : Status of display. Set by applications. Set by UCMC if applications not reached 0 = OK. 1 = KO.

- UcCardInside : Card presence at the end of transaction. Set by applications. Set by UCMC if applications not reached 0 = no card 1 =card present.
- UcMode : Set by applications to ucTrsMode of debit request. Information between applications and DLL protocol.
- UcFunction : Set by applications to ucTrsMode of debit request. Information between applications and DLL protocol. Allow to know the sub-structure to use in sCard field.
- UcTypeCardStruct : Set by applications to UCMHOST_CARD_UNKNOWN , UCMHOST_CARD_EMV or UCMHOST_CARD_BO or UCMHOST_CARD_MONEO. Used to indicate which structure to use; Convention between applications and protocol DLL. Not used by UCMC. Set by UCMC to UCMHOST_CARD_UNKNOWN if applications not reached.
- UcSupport : Set by applications to CHIP_SUPPORT or ucTrsSupport of debit request. Set by UCMC to ucSupport of debit request if applications not reached.
- UsAppName : Set by applications to SEGMENT number given by manager (debit_emv(SEGMENT, …)
- TAppLibelle : Set by applications. Application name.
- UlAmount : Set by applications using ulAmount of debit request.
- TCurrency : Set by applications using tCurrency of debit request.
- UcCardHolderLanguage : Set by applications to 0xFF. Not used by UCMC.
- Sloc : Used only by applications managed local record transaction.
- SCard : Set by applications. Convention between applications and DLL protocol.

6.3 T_UCMHOST_ DEVICE STRUCTURE

```
typedef struct
{
  unsigned char ucType ;
    unsigned char ucMode ;
    unsigned char ucPilote ;
    unsigned char uctNomDriver[ UCM_NAME_DLL_DRIVER_LG ] ;   /* Dll protocole
name */
    unsigned char uctNomDll   [ UCM_NAME_DLL_DRIVER_LG ] ;   /* Dll protocole
name */
    unsigned char ucCom ;       /* Port com number     */
    union
    {
          unsigned char         uctData [ UCM_MAX_SIZE_DATA_DEVICE ];
          T_UCM_ICC_DATA        sIcc;
          T_UCM_DA_DATA              sDa ;
                T_UCM_PPAD_DATA      sPpad;
          T_UCM_MONNAYEUR_DATA sMonnayeur ;
                T_UCM_COM_DATA       sComP;
     } u;

} T_UCM_DEVICE ;
```

Information used by DLL are :

- UcCom: give the COM number to open to dialog with Host.

The other information are used by UCMC.

6.4 T_UCMHOST_READ_TRACK

```
typedef struct
{
      int                     iStatus;
      TRACK1_BUFFER           track1;     // track 1 contents
      TRACK2_BUFFER           track2;     // track 2 contents
      TRACK3_BUFFER           track3;     // track 3 contents
} T_UCMHOST_READ_TRACK;
```

Track read depends on device.
Field iStatus can take following values:

- UCMHOST_TRACK_OK: ok read track (s).
- UCMHOST_TRACK_BUSY: Device busy
- UCMHOST_TRACK_MTNC : UCMC in maintenance mode.
- UCMHOST_TRACK_NO_CARD : Waiting time out. No Card.
- UCMHOST_TRACK_LUHN: Bad Luhn key code.
- UCMHOST_TRACK_KO : Bad track (parity, digits ...).
- UCMHOST_TRACK_NULL: Track is empty.
- UCMHOST_TRACK_DENIED : Reading denied: the track is known by an application
- UCMHOST_TRACK_ICC: Reading only.
- UCMHOST_TRACK_CANCEL: Operation is cancelled
- UCMHOST_TRACK_SEP: Separator error
- UCMHOST_TRACK_NUM: Numeric error
- UCMHOST_TRACK_LRC: LRC error
- UCMHOST_TRACK_PAR: Parity error
- UCMHOST_TRACK_NO_READER: No reader device defined.

6.5 T_UCMHOST_DEM_FCTAPP

```
typedef struct
{
      unsigned char ucFunction ;
      unsigned short usAppliNumber ; /* UCMHOST_APP_ALL for all or application_type
      given by object_info_t */

       union
      {
            unsigned char ucData[ 20 ];
            unsigned char ucRecord[ 12 ];
      } u;

} T_UCMHOST_DEM_FCTAPP;
```

Definition of each field:

- UcFunction: Describe service:
  - UCMHOST_FCTAPP_TLC_STATUS : Banking transaction downloading status request.
  - UCMHOST_FCTAPP_TLC_START : Banking transaction downloading start.
  - UCMHOST_FCTAPP_TLP_STATUS : Banking parameters downloading state
  - UCMHOST_FCTAPP_TLP_START : Banking parameters downloading start

- UCMHOST_FCTAPP_CONSULT : Banking applications consultation
- UCMHOST_FCTAPP_NB_RECORD_LOC : Specific for Banking applications managing local transaction records
- UCMHOST_FCTAPP_INFO_RECORD_LOC: Specific for Banking applications managing local transaction records
- UCMHOST_FCTAPP_LIST_RECORD_LOC: Specific for Banking applications managing local transaction records

- UsAppliNumber : Must be filled with UCMHOST_APP_ALL to reach all applications or by application_type of object_info_t structure.
- u.ucData[20] : RUF
- u. ucRecord[12] : Specific for Banking applications managing local transaction records

6.6 T_UCMHOST_FCTAPP

```
typedef struct
{
      unsigned char ucCr ;
      unsigned char ucRuf ;
      unsigned char ucFunction ;
      unsigned char ucNbAppli ;

      NO_SERIE     tSerial ;

      union
      {
            T_UCMHOST_APP_TLC_STATE sTLC [ UCMHOST_NB_MAX_APPLI ] ;
            T_UCMHOST_APP_TLC_START sTLCs [ UCMHOST_NB_MAX_APPLI ] ;
            T_UCMHOST_APP_TLP_STATE sTLP [ UCMHOST_NB_MAX_APPLI ] ;
            T_UCMHOST_APP_CONSULT   sAppCo [ UCMHOST_NB_MAX_APPLI ] ;
            T_UCMHOST_APP_NB_RECORD_LOC   sNbRecordLoc [ UCMHOST_NB_MAX_APPLI_RECORD
            ] ;
            T_UCMHOST_APP_LIST_RECORD_LOC sListRecordLoc [
            UCMHOST_NB_MAX_APPLI_RECORD ] ;
            T_UCMHOST_APP_INFO_RECORD_LOC sInfoRecordLoc ;
      } u;

} T_UCMHOST_FCTAPP;
```

Definition of each field:

- ucCr : Result of service
  - UCMHOST_FCTAPP_CR_OK : Response from application
  - UCMHOST_FCTAPP_CR_KO: Service no allowed at this moment.
  - UCMHOST_FCTAPP_CR_KO_MTNC: UCMC in maintenance mode.
  - UCMHOST_FCTAPP_CR_KO_NO_SERVICE: No response.
  - UCMHOST_FCTAPP_CR_KO_GETINFO_KO: Application identification error.
- UcRuf : RUF
- UcFunction: Describe service:
  - UCMHOST_FCTAPP_TLC_R_STATUS: Banking transaction downloading status result. Use T_UCMHOST_APP_TLC_STATE structure.
  - UCMHOST_FCTAPP_TLC_R_START : Banking transaction downloading status result. Use T_UCMHOST_APP_TLC_START structure.
  - UCMHOST_FCTAPP_TLP_R_STATUS : Banking parameters downloading status result. Use T_UCMHOST_APP_TLP_STATE structure. UCMHOST_FCTAPP_TLP_R_START : Banking parameters downloading start result. Use T_UCMHOST_APP_TLP_START structure.
  - UCMHOST_FCTAPP_CONSULT_R : Banking applications consultation result. Use T_UCMHOST_APP_CONSULT structure.
  - UCMHOST_FCTAPP_NB_RECORD_LOC_R : Specific for Banking applications managing local transaction records.
  - UCMHOST_FCTAPP_INFO_RECORD_LOC_R: Specific for Banking applications managing local transaction records.
  - UCMHOST_FCTAPP_LIST_RECORD_LOC_R: Specific for Banking applications managing local transaction records.
- UcNbAppli : Number of Applications responses.
- TSerial: CAD30 serial number.

6.7 T_UCMHOST_APP_TLC_STATE

```
typedef struct
{
      object_info_t     sInfo ;
      unsigned char   ucState ;
      unsigned char   ucRuf ;

      union
      {
            T_UCMHOST_FCTAPP_ACCEPT sAccept;
            unsigned char ucBuf[ UCMHOST_MAX_SIZE_FCTAPP_ACCEPT_INFO];
      }u;

      unsigned char ucNbTrs;   /* */

      unsigned char ucRuf2;   /* */

      union
      {
            T_UCMHOST_REMISE_CB sCB[UCMHOST_MAX_FCTAPP_MAX_TRS_INFO];

            unsigned char ucBuf[
            UCMHOST_MAX_SIZE_FCTAPP_TRS_INFO*UCMHOST_MAX_FCTAPP_MAX_TRS_INFO];
      }uTrs ;

} T_UCMHOST_APP_TLC_STATE;
```

Definition of each field:

- sInfo : See object_info_t.
- ucState : State
  - UCMHOST_TLC_STATE_OK: Downloading complete.
  - UCMHOST_TLC_STATE_NOT_COMPLET: Last downloading not complete.
  - UCMHOST_TLC_STATE_CALL_ERROR: Error calling service of applications
  - UCMHOST_TLC_STATE_NO_CALL: No response from application.
- ucRuf : RUF
- ucNbTrs : Transaction number. Must be less than UCMHOST_MAX_SIZE_FCTAPP_MAX_TRS_INFO
- ucRuf2 : RUF
- u.sAccept : See banking applications.
- uTrs.sCB : See banking applications.

6.8 T_UCMHOST_APP_TLP_STATE

```
typedef struct
{
      object_info_t     sInfo ;
      unsigned char   ucState ;
      unsigned char   ucRuf ;

      union
      {
            T_UCMHOST_FCTAPP_ACCEPT sAccept;
            unsigned char ucBuf[ UCMHOST_MAX_SIZE_FCTAPP_ACCEPT_INFO];
      }u;
      unsigned char ucAppInit;   /* Not initialized = 0  Initialized = 1 */
      unsigned char ucAppActive; /* Not active = 0     Active = 1 */

      unsigned char ucTLCError[ 4 ];      /* Telecollecte */
      unsigned char ucTLCErrorTable[ 2 ]; /* Table */
      unsigned char ucTLCErrorCnct[ 2 ];  /* Connection */

      unsigned char ucNbTable;

      unsigned char ucRuf2;   /* */
      unsigned char ucRuf3;   /* */

      union
      {
            T_UCMHOST_TLP_TABLE stab[UCMHOST_MAX_FCTAPP_MAX_TABLE_INFO];

            unsigned char ucBuf[
            UCMHOST_MAX_SIZE_FCTAPP_TABLE_INFO*UCMHOST_MAX_FCTAPP_MAX_TABLE_INFO];
      }uTable ;

} T_UCMHOST_APP_TLP_STATE;
```

Definition of each field:

- sInfo : See object_info_t.
- ucState : State
  - UCMHOST_TLC_STATE_OK: Downloading complete
  - UCMHOST_TLC_STATE_NOT_COMPLET: Last downloading not complete.
  - UCMHOST_TLC_STATE_CALL_ERROR: Error calling service of application
  - UCMHOST_TLC_STATE_NO_CALL: No response from applications.
- ucRuf : RUF
- u.sAccept : See banking applications.
- ucAppInit : Application not initialized = 0, Initialized = 1
- ucAppActive : Application not active = 0, Active = 1
- ucTLCError : See banking applications.
- UcTLCErrorTable : See banking applications.
- UcTLCErrorCnct : See banking applications.
- UcNbTable: Transaction number (less than UCMHOST_MAX_FCTAPP_MAX_TABLE_INFO).
- ucRuf2 : RUF
- ucRuf3 : RUF
- uTable.Stab : See banking applications.

6.9 T_UCMHOST_APP_TLC_START

```
typedef struct
{
      object_info_t     sInfo ;
      unsigned char   ucState ;
      unsigned char   ucRuf ;
} T_UCMHOST_APP_TLC_START;
```

Definition of each field:

- sInfo : See object_info_t.
- ucState : State
  - UCMHOST_TLC_STATE_OK: Downloading complete
  - UCMHOST_TLC_STATE_NOT_COMPLET: Last downloading not complete.
  - UCMHOST_TLC_STATE_CALL_ERROR: Error calling service of application
  - UCMHOST_TLC_STATE_NO_CALL: No response from applications.
- ucRuf : RUF

6.10 T_UCMHOST_CONSO

```
typedef struct
{
      object_info_t           sInfo ;

      unsigned char   ucAppStatus ;
      unsigned char   ucFileStatus ;

      unsigned char   ucRuf[ 20 ] ;

}     T_UCMHOST_APP_CONSULT ;
```

Definition of each field:

- sInfo : See object_info_t.
- ucAppStatus : Application state
  - UCMHOST_APP_STATUS_INIT_ACTIVE: Application initialized and active
  - UCMHOST_APP_STATUS_INIT_ACTIVE: Application initialized not active.
  - UCMHOST_APP_STATUS_NOT_INIT_ACTIVE: Application not initialized, active.
  - UCMHOST_APP_STATUS_NOT_INIT_NOT_ACTIVE: Application not initialized not active.
  - UCMHOST_APP_STATUS_NOT_SIGNED: Application not signed.
- UcFileStatus : File state
  - UCMHOST_APP_FILE_EMPTY:
  - UCMHOST_APP_FILE_NOT_EMPTY:
  - UCMHOST_APP_FILE_FULL:
  - UCMHOST_APP_FILE_NOT_SIGNED
- ucRuf : RUF

6.11 T_UCMHOST_D_CONNECT

```
typedef struct
{
      unsigned char ucNetwork;
      union
      {
            T_UCMHOST_X25     hostX25;
            unsigned char ucData[ 200 ] ;
      }u;
} T_UCMHOST_D_CONNECT;
```

Definition of each field:

- ucNetwork : Set by manager. Only UCMHOST_NET_X25. Use T_UCMHOST_X25 structure.
- u.hostX25 : See Network structure.

Network structure.

```
typedef struct
{
      unsigned char ucTypeProt; /* STR_ETABL_CONNEX protocol.See ccext.h from sdk */
      unsigned char ucTypePad;    /* STR_ETABL_CONNEX type_PAD EBA / EMA */
      unsigned char ucTypeCentre; /* Reason of call */
      unsigned char ucLgComplX25; /* length of uctComplX25 */
      unsigned char uctComplX25[ UCMHOST_LG_ADR_COMPLX25 ];
      unsigned char uctRaccord[ UCMHOST_LG_ADR_RACCORD ];
      unsigned char ucLgRaccord;
      unsigned char ucLgAppel;
      unsigned char uctAppel[ UCMHOST_LG_ADR_APPEL ];
      unsigned char ucTimer;
      unsigned char ucRuf[ 40 ] ;
}T_UCMHOST_X25 ;
```

Definition of each field:

- ucTypeProt : Type of protocol. Set by manager. See STR_ETABL_CONNEX.
- ucTypePad : Type of PAD (EBA or EMA). Set by manager. See STR_ETABL_CONNEX.
- ucTypeCentre : Reason of call (Authorization or downloading). Set by manager. See cb2a_cmp.h from sdk (exemple SERVICE_TELECOLLECTE…).
- ucLgComplX25 : Set by manger. Same value as field lgr_data_compl of STR_ETABL_CONNEX structure.
- uctComplX25[UCMHOST_LG_ADR_COMPLX25]: Set by manager. Same value as field data_compl_X25 of STR_COMP_X25 structure.
- uctRaccord[UCMHOST_LG_ADR_RACCORD]: Set by manager. Same value as field adr_raccord of STR_ETABL_CONNEX structure.
- UcLgRaccord : Set by manager. Same value as field lgr_adr_raccord of STR_ETABL_CONNEX structure.
- UcLgAppel: Set by manager. Same value as field lgr_adr_appel of STR_ETABL_CONNEX structure.
- uctAppel[UCMHOST_LG_ADR_APPEL]: Set by manager. Same value as field adr_appel of STR_ETABL_CONNEX structure.
- ucTimer : Set by manager. Same value as field timer_TNR of STR_ETABL_CONNEX structure.
- ucRuf[40] : RUF

6.12 T_UCMHOST_R_CONNECT

```
typedef struct
{
      unsigned char ucStatusCn;
      unsigned char ucStatusHost;   /* RUF */

      union
      {
            unsigned char ucData[ 50 ] ;  /* RUF */
      }u;

} T_UCMHOST_R_CONNECT;
```

Definition of each field:

- ucStatusCn : Status of connection. Set by DLL protocol.
  - UCMHOST_CN_OK :
  - UCMHOST_CN_KO :
  - UCMHOST_CN_PBX25 :
  - UCMHOST_CN_CANCEL :
  - UCMHOST_CN_NODIALTONE :
  - UCMHOST_CN_NOCARRIER :
  - UCMHOST_CN_HANGUP :
  - UCMHOST_CN_BUSY :
  - UCMHOST_CN_BLIND
  - UCMHOST_CN_NOANSWER
- ucStatusHost : State of Host. RUF
- u.ucData[50] : RUF

6.13 T_UCMHOST_STATUS_UCM

```
typedef struct
{
      unsigned char ucUCM;       /* example UCMHOST_STATE_MAINTENANCE */
      unsigned char ucM2OS;      /* manager state */
      unsigned char ucICC;       /* Card present = UCMHOST_ICC_IN */
      unsigned char ucNetwork;   /*  */
      unsigned char ucPrinter;   /* HS or no paper */
      unsigned char ucPinpad;    /* OK = 0 */
      unsigned char ucDisplay;   /* OK = 0 */
      unsigned char ucDevice;    /* Other device 0=Ok else pb */
} T_UCMHOST_STATUS_UCM;
```

Definition of each field:

- ucUCM : Status of connection.
  - UCMHOST_STATE_IDLE :
  - UCMHOST_STATE_START : Starting phase
  - UCMHOST_STATE_BUSY :
  - UCMHOST_STATE_MAINTENANCE : CADTOOL device connected
  - UCMHOST_STATE_TLC : Banking transaction downloading
  - UCMHOST_STATE_TLP: Banking parameters downloading
  - UCMHOST_STATE_DOWNLOAD: Software downloading. RUF.
  - UCMHOST_UCMHOST_STATE_HS: Device or UCM out of order.
- ucM2OS : Manager phase. See manager documentation
- ucICC : Icc state.
  - UCMHOST_ICC_OUT : Card outside ICC. OK.
  - UCMHOST_ICC_IN : Card inside. ICC OK.
  - UCMHOST_ICC_HS and other value: ICC HS.
- ucNetwork : Network state.
  - UCMHOST_NETWORK_OK :
  - UCMHOST_NETWORK_KO :
  - UCMHOST_NETWORK_START :
  - UCMHOST_NETWORK_ONLINE :
- ucPrinter : Printer state.
  - UCMHOST_PAPER_OK:
  - UCMHOST_PAPER_KO:
  - UCMHOST_PRINTER_KO:
- ucPinpad : Pinpad state. 0 = OK.
- UcDisplay : Display state. 0 = 0K.
- ucDevice : Other device state. UCMHOST_DEVICE_OK if OK.

6.14 T_UCMHOST_MPA_STATUS

```
typedef struct {
      unsigned char ucDLL;         /* Status DLL = Init finished ... */
      unsigned char ucCom;         /* Status communication = No com ...*/
      int           iNbErrSend;    /* */
      int           iNbErrRece;    /* */
      int           iNbMsgNotSent; /* Msg not sent */
      int           iNbMsgSent;    /*  */
      int           iNbMsgRece;    /*  */
   char          cRuf[ 40 ];
}T_UCMHOST_MPA_STATUS;
```

Definition of each field:

- ucDLL : Status of DLL. Set by DLL protocol.
  - UCMHOST_DLL_IDLE:
  - UCMHOST_DLL_SESSION : vend session
  - UCMHOST_DLL_NOINIT :
  - UCMHOST_DLL_INHIBITED :
  - UCMHOST_DLL_INITKO :
  - UCMHOST_DLL_KO : Internal error
- ucCom : Status of communication
  - UCMHOST_MPA_COM :
  - UCMHOST_MPA_NOCOM :
  - UCMHOST_MPA_NOCOM_NOW : Com has been established but is unaivalable
- iNbErrSend : Number of emission errors.
- iNbErrRece : Number of reception errors.
- iNbMsgNotSend : Number of messages not sent.
- iNbMsgSent : Number of messages sent.
- iNbMsgRece : Number of messages received.
- CRuf : Ruf

7. DLL MEMORY DECLARATION

Memory used by DLL is described in SDK sample HOTESDK.

```
ADD_ROM    = E0900000
ADD_RAM    = E0980000
SIZE_ROM    = 512K
SIZE_RAM    = 512K
```