Connect:Express® Unix

Release Notes

Version 1.4.6-1

Sterling Commerce
An IBM Company

Table of Contents

28/01/2010 : **SR 15973** : problem fixed in purge_queue.sh. on SUN SOLARIS (tom/config).
purge_queue.sh updated.

02/02/2010 : **SR 15943** : New procedure check_httpn.sh added to control that the process tom_httpn is active and restart it if required (use crontab to activate it periodically).
(tom/httpn).
survey_httpn.sh added.

08/02/2010 : **case 135892** : New procedure check_apm.sh added to control that the process tom_apm is active and listening to a tcp/ip port, and restart it if required (use crontab to activate it periodically). (tom/config).
survey_apm.sh added..

10/02/2010 : **case 138833** : tom_prm returns code 1 in any case (when no parameter file is provided). (tom/prm).
tom_prm updated.

17/02/2010 : **case 143439** : New environment variable added $THR_PI42, to place in the .profile file. This variable is used when receiving the file size from the sender : it determines the error threshold (%) Connect:Express will apply to the file size when controlling the reception (Pi42 peSIT parameter).
Change in decl.h, declc.h and tom_apm.c (tom/strf).
tom_apm updated.

24/02/2010 : **case 133791** : Default values changed in the presentation and session tables (message size 4096, multi-article option = N)
Change in oto0.c (tom/gtrf).
tom_mon, RPRE and RTAB updated.

10/03/2010 : **case 152743** : In server mode, version 145 added a control of the first mesage received. The length should be 24 characters for PeSIT. Some PeSIT softwares are sending 26 characters (for some reason). The control is made more flexible .
Change in r_tcp.c (tom/strf).
tom_apm updated.

13/04/2010 : **case 164852** : New environment variable added $REC_EMPTY, to place in the .profile file. If the variable $REC_EMPTY exists and is equal to 1, empty records (length = 0) in text variable files (TV) are replaced by one space character (lg = 1).
Change in decl.h, declc.h and tom_apm.c (tom/strf).
tom_apm updated.

Version 1.4.6 provides new PeSIT services, new SSL functionality and new FTP identification facility. These functionalities are described below and updates are included in the respective documentation.

PeSIT message and store and forward are included into the *Connect:Express Unix User and Installation Guide* along with the general information about new functionalities. The control of certificates is included into the *Connect:Express Unix SSL Option Guide*. The FTP extended identification is included into the *Connect:Express Unix FTP Guide*. The description of the p1b8pe2e utility is included into the *Connect:Express Unix User and Installation Guide.*

PeSIT Message

This section provides information on what PeSIT Message is, how to use it and how to configure it in Connect:Express Unix.

Overview

PeSIT Message is a protocol feature that enables to send data in one step :

```
Message = data <-> AckMessage
```

instead of :

```
       Create <-> AckCreate
                  +
         Open <-> AckOpen
                  +
        Write <-> AckWrite
                  +

                …….
             N * Data
                  /
          Sync <-> Async
                …….

                  +
              DataEnd
                  +
     TransEnd <-> AckTransEnd
                  +
        Close <-> AckClose
                  +
    Deselect <-> AckDeselect.
```

You can use this feature to send short messages or files, and also to perform end to end acknowledgement either in a standard file transfer process or in a store and forward process. The Store and forward process is described in next section, including the end to end acknowledgment. A new utility, called p1b8pe2e is provided to send end to end acknowledgement or to forward files or messages.

PeSIT Message Operational Characteristics

This section describes PeSIT message processes.

To send a message
To receive a message

Sending a Message - Type of Request M

The user can send a message without data or with data, using either the "*P99*" - or "*USER DATA*" - field or a file to pass them. The type of request 'M' indicates that this is a message. If the "*P99*" field is provided, this is the data to send. If no "*P99"* field is provided and a "*DSN"* value is provided, the data is sent from the file. The parameters of the message transfer request provide a symbolic file name: if this name is defined in the directory (RFIC), the definition is used. If this name doesn't exist, the $$MSGD$$ definition is looked for: if found, and status enabled, it is used. If $MSGD$$ is not defined, or status disabled, the request is rejected.

The size of the message unit will be determined by the record length if it is provided in the symbolic file definition or in the request parameters, or the session message length, with a maximum of 4096 characters.

In the following the various interfaces are shown: the only parameter to consider is the type of request. All other parameters are similar to any other type of transfer request. The physical file name is not required.

STERM - Set TYPE field to 'M' and provide data, if needed, using PHYSICAL NAME or USER DATA.

```
C:E/UNIX 146-100 ----------- TRANSFER REQUEST     ----------------------- tom1
OPTION ===>
FILE .................. :                DIRECTION ............ :   (T/R)
PARTNER ............... :
DPCSID ALIAS .......... :                DPCPSW ALIAS .......... :
ORIGIN ................ :                DESTINATION ........... :
SENDER  :                                RECEIVER  :
PHYSICAL NAME ........ :
USER DATA ........... :
LABEL :

RECORD FORMAT ......... :                TF, TV, BF, BU
RECORD LENGTH ......... :
TYPE/STRUCTURE/MODE FTP :                E/A/I/*,F/R/*,B/S/*
STORE UNIQUE (FTP) .... :                Y/N  FA :   Y/N   NOT :   (0-7)
TYPE ................ : M                (N/I/H/M)
TYPE OF CONNECTION .... :                (X/P/T)
PRIORITY ............. :                 (0/1/2)
DATE .................. : 20100407135855  (YYYYMMDDHHMMSS)
API FIELD (ETEBAC3 : 80 CHARACTERS FOR CARD)
1...5....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0


-ENTER- NEXT FIELD                -F3- CANCEL                 -F8- COMPLETION
```

p1b8preq : Set parameter `/TYP=M` and use /P99= or /DSN= to provide data

API Z20 : Set field char typ[1]; /* Request type = M */
Provide data in char s_pi99_254[254]; /* Sender Pi99 */
char dsnam[44]; /* File physical name */

Receiving a Message –Saving Data

When receiving data with the Message service, Connect:Express uses the symbolic file name from the PeSIT parameter Pi12: if this name is defined in the directory (RFIC), the definition is used. If this name doesn't exist, the $$MSGD$$ definition is looked for: if found it is used, if not found the request is rejected.

A message can carry either data, or an end to end acknowledgment of a previous file transfer: the PeSIT parameter Pi11 indicates if this is a data Message (hexadecimal 'FFFF' is for initial message, 'FFFE' is for message acknowledgment) or an end to end acknowledgment Message (Pi11 is the same as the original CREATE parameter).
There are two possibilities for storing the data of the Message: writing it into a file or saving it into the RENC file. Connect:Express will decide where to store it from the file attributes of the file definition. If a physical file name is provided in the file definition, Connect:Express will store data in a file. Connect:Express will place first 254 characters of data in the r_pi99_254 field of the RENC file , displayed in MESSAGE << field of STERM.

```
10/06/04 16:24:23 REQUEST 07200008 MSGFIC2 <- BOUCLE MESSAGE ACCEPTED  STRF 0000011700
10/06/04 16:24:23 REQUEST 07200008 MSGFIC2 <- BOUCLE MESSAGE RECEIVED  STRF 0000011700
10/06/04 16:24:23 REQUEST 07200008 254 first characters of the file
10/06/04 16:24:23 REQUEST 07200008 $TOM_DIR/msg/MSGFIC2_A7200008
```

If no physical file name is provided (the field must be set to '-') , the data will be considered as user data and placed in the r_pi99_254 field of the RENC file (254 characters maximum).

```
10/06/04 16:24:23 REQUEST 07200008 MSGFIC2 <- BOUCLE MESSAGE ACCEPTED  STRF 0000011700
10/06/04 16:24:23 REQUEST 07200008 MSGFIC2 <- BOUCLE MESSAGE RECEIVED  STRF 0000011700
10/06/04 16:24:23 REQUEST 07200008 254 first characters of the file
```

STERM monitoring screens show the message information.

```
C:E/UNIX 146-1 ----------------- MONITOR STATUS --------------------------- ce01
OPTION ===>
        REQ.NUM.   FILE     WITH       DIR.  PRI. REQ. TYPE STATE  STRF ID

        07200001   FICTEST1 EXPRESS1   T     0    N NORMAL  E      0000010408
        07200005   FICTEST2 DPX1       T     0    N NORMAL  E      0000011441
        07200006   FICTST   SID1       R     0    N NORMAL  E      0000011698
        07200007   FICTEST2 DPX1       T     0    N NORMAL  E      0000011443
        07200008   MSGFIC2  BOUCLE     R     0    M MESSAGE E      0000011700
        07200009   FICSTSN  DPX1       T     0    N NORMAL  E      0000011445
        07200010   FIC22424 SID1       R     0    N NORMAL  E      0000011702
        07200011   FICTEST3 DPX1       T     0    N NORMAL  E      0000011447
        07200012   ARECEVOI SID1       R     0    N NORMAL  E      0000011704
        07200013   FICTEST3 DPX1       T     0    N NORMAL  E      0000011449
        07200014   ARECEVOI SID1       R     0    N NORMAL  E      0000011706
        07200015   AENVOYER DPX1       T     0    N NORMAL  O      0000011451
        07200017   FICTEST4 DPX1       T     0    N NORMAL  E      0000011456
        07200018   FICTST2  SID1       R     0    N NORMAL  E      0000012225
        07200019   FICTEST1 EXPRESS1   T     0    N NORMAL  J      0000011458


       <- -F10- -F3- END -F7- PREVIOUS SCREEN -F8- NEXT SCREEN -F11- ->
```

```
C:E/UNIX 146-1 ---------------- MONITOR STATUS -------------------------- tom1

  REQUEST : 07200008    FROM : BOUCLE       DIRECTION : R     WITH : BOUCLE
  ORIGIN  : BOUCLE  DESTINATION : BOUCLE  XFER ID : 06292271  *MESSAGE
  SENDER  :                            RECEIVER  :
  USERID  : gcz         STRF PID    : 0000014700  FA : N    NOT : 0    SSL : N
  FILE    : MSGFIC2     CMD ORIGIN  : I  SSLPARM :          TRANSFER STATE : X
  PROTOCOL: PESIT  CRC : N    MULTI : N   TRANSLATION : 0 COMPRESSION : 0
  DSNAME  : $TOM_DIR/msg/MSGFIC2_A7200008
  MESSAGE << :  254 first characters of the file



  BEGIN : 20100406 16:24:23  END : 20100406 16:24:23      RETRIES/MAX : 00/00
  NRC : 0000    SRC : 0000   TRC : 0000   PRC : 0 000     SSLRC : 00000000
  NUMBER OF RECORDS XFERED : 0000000005      K.BYTES    :  0000000019
  RECORD FORMAT ......... : BU               RECORD LENGTH ......... : 04096
  ********** TCPIP **********

  TCPIP HOST (09) : localhost
  TCPIP ADDRESS (00) :
  PORT : 05015
```

Store and Forward – End to End Acknowledgment

This section provides information on the store and forward process (with end to end acknowledgment based on PeSIT Message), how to use it and how to configure it in Connect:Express Unix. First, the end to end acknowledgment is described, as a simple process, then the more complicate store and forward associated with end to end acknowledgment is described.

Overview

In the following we use the acronym “EERP” for “End to End ResPonse”. The end to end response acknowledges that a file (or a message) has been received by the destination application. This can be a simple acknowledgment from receiver to sender, or a store and forward acknowledgment, from final destination to initial origin.

End to End Acknowledgment - EERP

This section describes the different steps of the end to end process. Next section shows how it can be integrated into the store and forward mechanics.

- o Step one : At end of reception, save end to end context, including the transfer id.
- o Step two : Retrieve end to end context and submit an EERP transmission request with it.
- o Step three : Receive the end to end acknowledgment and take appropriate action.

The type of request is provided in the parameter list of the user command ($25), to enable user to take actions specific to a file (Type=N or I), a message (Type=M), or an EERP (Type=E). All parameters required to identify a transfer , the end to end context, are provided in the parameter list of the command. Parameters required to set up the store and forward process (for example to save the EERP context for further acknowledgment) are listed below.

Saving Parameters

Step one is normally done through the RENC file, where all end to end parameters are saved. The end of transfer command enables you to save these parameters, or to use them on line. The table below shows the relationship between Connect:Express parameters, PeSIT parameters, RENC file fields, the normal transfer request parameters and user command fields.

Parameter saved	PeSIT-Pi	RENC-trfpar	Normal Request	User command
Partner Identification	3	pi.ident	SPN=	$3
Alias	4	pi.idser	SID=	$13
EERP Context				
File identification	3bis	pi.user_org	ORG=	$11
	4bis	pi.user_dst	DST=	$12
	11	pi.tyf		$26 (new)
	12	pi.nof	SFN=	$2
Transfer identification	13	pi.idt		$27 (new)
File attribut	51	pi.dhc		$28 (new)
Sender identification	61	pi.user_snd	SND=	$15
Receiver identification	62	pi.user_rcv	RCV=	$16

The EERP transfer process must use access to RENC to build the Message data unit, unless these parameters are provided directly to the end to end utility called **p1b8pe2e**, that is described below in "The *End to End Utility P1b8e2e*" .

Retrieving Parameters

Step two must build the EERP transfer request parameter list with information required to retrieve the end to end context, and an optional user message to associate with the context. There are two possibilities: to give the request number or to provide all parameters.

o Giving the request number and an optional user message – if the request is ended and recorded in the RENC file.

Information Expected	Field	Description	PeSIT
Request number	REQ=	The request (local) to acknowledge	
Partner Identification	SPN=	Where to send it (default = partner)	Pi3 (Connect)
Local Identification	SID=	My name (default = alias)	Pi4 (Connect)
Notification	NTF=	0-7	
Priority	PRT=	0-2	
Link type	LNK=		
Scheduling date	DAT=		
User message	ACK=	Provides feedback, <= 254	Pi91

o Giving the end to end parameters and the user message.

Information Expected	Field	Description	PeSIT
Partner Identification	SPN=	Where to send it	Pi3 (Connect)
Local Identification	SID=	My name (default = alias)	Pi4 (Connect)
Notification	NTF=	0-7	
Priority	PRT=	0-2	
Link type	LNK=		
Scheduling date	DAT=		
EERP Context			
File identification	ORG=	L<=24	Pi3bis
	DST=	L<=24	Pi4bis
	P11=	L = 2 hexadecimal	Pi11
	P12=	L<=14	Pi12
Transfer identification	P13=	L<=08 decimal	Pi13
File attribute	P51=	L=12	Pi51
Sender identification	P61=	L<=24	Pi61
Receiver identification	P62=	L<=24	Pi62
Feedback			
User message	ACK=	Provides feedback lg <= 254	Pi91

Sending End to End Response

To send the acknowledgment, the user must submit an EERP transfer request to Connect:Express , using the batch utility p1b8e2e, a program, or the operator interface STERM. Connect:Express builds the EERP message from the EERP context, either from the parameters provided, or accessing to the RENC file. The initial request must be a reception, a file or a message, with status ended = ‘E’. TRC=2050, 2051, 2053 or 2055 is issued if the request is in the RENC file and it does not meet the conditions.

The EERP process doesn’t require a file definition to execute: if symbolic file $$EERP$$ is defined, and status enabled, the process will be executed according to this profile, in any case: for example, exits, commands, physical file name attached to this profile are used. If the file provided in the request is defined and no $$EERP$$ definition exists, or status is disabled, the transfer will be executed according to the file of the request.

Upon reception of an EERP, Connect:Express searches for the corresponding request. It must be a transmission, of a file or a message, and it must be Ended. TRC code 2050 or 2055 is issued if the request is found and doesn’t match these conditions. If the request doesn’t exist, the EERP is accepted.

The status of the request in the RENC file is changed from ‘E’ to ‘X’ when the corresponding EERP transfer is successfully completed.

Using STERM

You can submit an EERP request without feedback, using STERM. The EERP is built from information retrieved in the RENC file. No feedback information is provided is the PeSIT message.

```
C:E/UNIX 146-1 ----------------- MONITOR STATUS --------------------------- ce01
OPTION ===>
     REQ.NUM.   FILE       WITH       DIR.  PRI.  REQ. TYPE  STATE  STRF ID
     07200001   FICTEST1   EXPRESS1   T     0     N NORMAL   E      0000010408
     07200003   FICTEST1   EXPRESS1   T     0     E EERP     E      0000013254
     07200005   FICTEST2   DPX1       T     0     N NORMAL   E      0000011441
     07200006   FICTST     SID1       R     0     N NORMAL   E      0000011698
     07200007   FICTEST2   DPX1       T     0     N NORMAL   E      0000011443
     07200008   DOUDOU     SID1       T     0     M MESSAGE  E      0000011700
     07200009   FICSTSN    DPX1       T     0     N NORMAL   E      0000011445
     07200010   FIC22424   SID1       R     0     N NORMAL   E      0000011702
     07200011   FICTEST3   DPX1       T     0     N NORMAL   E      0000011447
     07200012   ARECEVOI   SID1       R     0     N NORMAL   E      0000011704
E    07200013   FICTEST3   DPX1       R     0     N NORMAL   E      0000011449
     07200014   ARECEVOI   SID1       R     0     N NORMAL   E      0000011706
     07200015   AENVOYER   DPX1       T     0     N NORMAL   O      0000011451
     07200017   FICTEST4   DPX1       T     0     N NORMAL   E      0000011456
     07200018   FICTST2    SID1       R     0     N NORMAL   E      0000012225
     07200019   FICTEST1   EXPRESS1   T     0     N NORMAL   J      0000011458

      <- -F10- -F3- END -F7- PREVIOUS SCREEN -F8- NEXT SCREEN -F11- ->
```

Using P1b8pe2e Utility

If you want to send a feedback message with the EERP, use the p1b8pe2e utility, with parameter /ACK=, or /DSN= if you want to place the feedback in a file.

FUN=E for 'send EERP', REQ='*request number*', ACK='*feedback message*': this will retreive information from the RENC file, and associate a feedback.

```
p1b8pe2e  "/FUN=E/SPN=ident/REQ=xxxxxxxx"   "/ACK='User Message'"
```

If the request is no longer in the RENC file, you will have to provide all information.
FUN=E, EERP context (/ORG=/DST=/P11=/P12=/P13=/P51=/P61=/P62=), ACK='*feedback message*'.

```
p1b8pe2e  "/FUN=E/SPN=ident"  "/'EERP context'" "/ACK='User Message'"
```

Using API L0b2z20

To submit an EERP request from a program, use d0b8z20.h as you would for a transfer request, and provide the specified information.

```
struct st_sci {
  char dire[1];                   /* Direction                         */
  char file[8];                   /* Symbolic file name      p1b8pe2e  */
  char part[8];                   /* Symbolic partner name   p1b8pe2e  */
  char dsnam[44];                 /* Dsname                            */
  char prty[1];                   /* Priority                          */
  char dat[8];                    /* Date                              */
  char hour[6];                   /* Hour                              */
  char lnk[1];                    /* Link type                         */
  char udf[44];                   /* User data file                    */
  char typ[1];                    /* Request type  = E p1b8pe2e        */
  char sta[1];                    /* State of Request                  */
  char dpcsid[8];                 /* Dpcsid for Alias                  */
  char dpcpsw[8];               /* Dpcpsw for Alias                  */
  char format[2];               /* Record Format (TF TV BF BU)       */
  char lrecl[5];                /* Record Length                     */
  char api[88];                 /* Api Field                         */
  char tsm[3];                  /* Type/Structure/Mode FTP           */
  char stou[1];                 /* Store Unique FTP                  */
  char fa[1];                   /* flag File agent Y/N               */
  char label[80];               /* Label                             */
  char s_pi99_254[254];         /* Feedback on 254         p1b8pe2e  */
  char user_org[8];             /* User Origin             p1b8pe2e  */
  char user_dst[8];             /* User Destination        p1b8pe2e  */
  char user_snd[24];            /* User Sender   pi61      p1b8pe2e  */
  char user_rcv[24];            /* User Receiver pi62      p1b8pe2e  */
  char quant_aa[2];             /* AA for Julian Date                */
  char quant[3];                /* Julian Date                       */
  char notif[1];                /* Notification: space/0-7           */
  char noreq[8];                /* request number          p1b8pe2e  */
  char dhc[12];                 /* File date Pi51          p1b8pe2e  */
  char idt[8];                  /* Pi13                    p1b8pe2e  */
  char ftype[4];                /* Pi11                    p1b8pe2e  */
  char filler[SIZE_RENC - 675];
};
```

Receiving End to End Response

Receiving an end to end response means that data is received through the PeSIT message service, Pi11 different from FFFF or FFFE. The file name is provided by Pi12. The process is similar to the PeSIT message process described before.

The EERP process doesn’t require a file definition to execute: if symbolic file $$EERP$$ is defined, and status enabled, the process will be executed according to this profile: for example, exits, commands, physical file name are used. If the file is defined and no $$EERP$$ definition exists, or status is disabled, the transfer will be executed according to this profile.

When receiving an EERP, Connect:Express searches for the corresponding request. The request must be a transmission, a file or a message, with status ended = ‘E’. TRC=2050 or 2055 is issued if the request does not meet the condition. If the request is not found, the EERP is accepted. The status of the corresponding request in the RENC file is changed from ‘E’ to ‘X’ when the EERP transfer is successfully completed.

```
09/07/03 16:00:22 REQUEST 00100026 FICMSG  <- partner EERP:  org    dest   idt
09/07/03 16:00:22 REQUEST 00100026 FICMSG  <- partner EERP    RECEIVED
09/07/03 16:00:22 REQUEST 00100026 User data
```

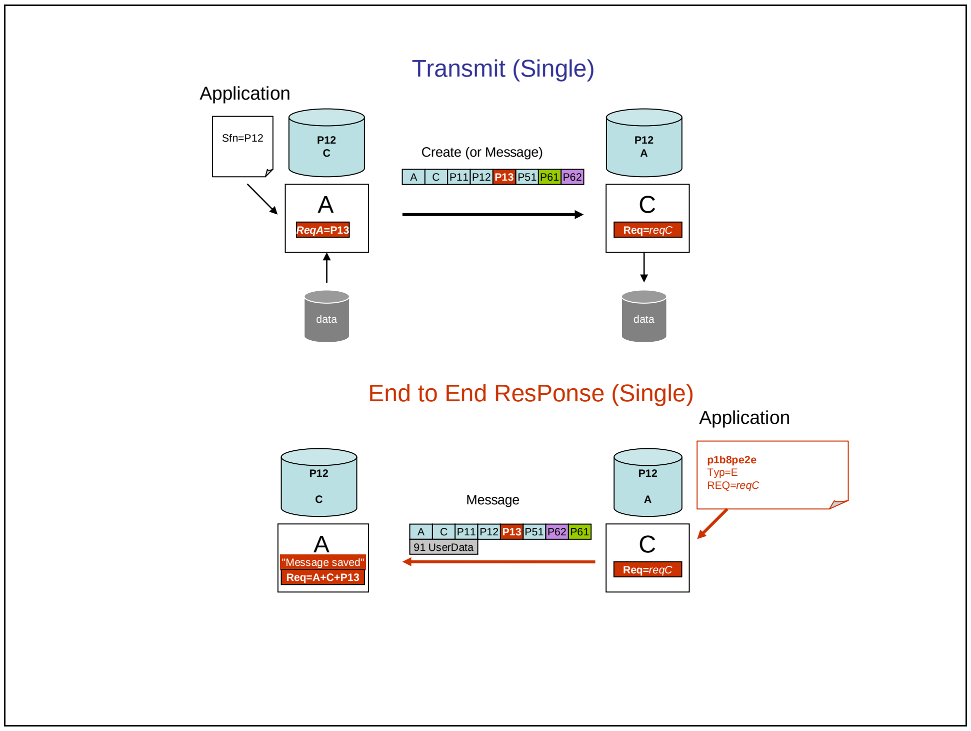
254 characters of the feedback message received are shown in STERM monitoring screens.

Store and Forward Operational Characteristics

The figure below shows how EERP works, in the most simple process: A sends a file to C, and C sends back an EERP to acknowledge reception.

The A request number, *ReqA*, is set in Pi13 that is the file transfer identification. C receives the file, with *ReqC* request number. The local *ReqC* record is saved. The application acknowledges the file using the *ReqC* information in which Pi13 has been saved. The end to end response is built from *ReqC* and sent in a PeSIT message to A. A receives the EERP message and checks in its RENC file the request that is being acknowledged from the information A+C+Pi13.

When the EERP is successfully sent, C changes *ReqC* status from E to X, and A changes *ReqA* status from E to X.



The message must be forwarded or not, depending if EERP is part of a store and forward process or not.

Overview

You can set up a store and forward process using p1b8pe2e utility. End of transfer commands enable you to save parameters for further use, or to activate automatic forwarding or acknowledgment. The store and forward function is available on Connect:Express Unix. User commands are provided to perform store and forward. Next section describes the automatic store and forward process.

Using Automatic Routing

When the DPCSID ALIAS field of the partner is set to ***xxxx*** - where '*xxxx*' is any string composed of A-Z, 0-9, a-z - the UEXxxxx command is launched at end of reception. You can use this mechanism to forward a file, a message or an EERP to the destination. The following store and forward user commands are provided:

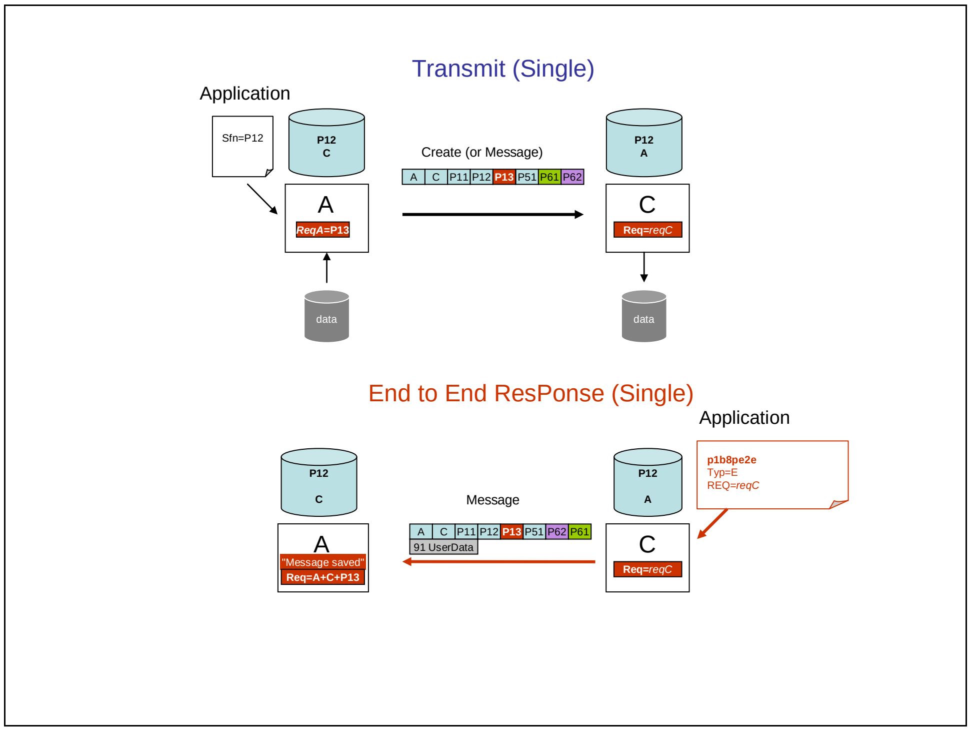
exit/UEXFWRD	uses p1b8pe2e utility to perform a PeSIT forward process.
exit/UEXEERP	uses p1b8pe2e utility to send an EERP.
exit/UEXROUT	uses p1b8preq utility to forward the file according to origin/destination (pi3bis/pi4bis).

The figure below shows that A is sending a file or a message to C, via B, and C is sending back the acknowledgment of the reception to A, via B.

Partner A and C are configured in B with DPCSID ALIAS = **FWRD**. B launches UEXFWRD at end of reception of any file or message from A, using p1b8pe2e with FUN=F, type of request = N or M. All end to end transfer PeSIT fields are forwarded in the new transfer. The transfer request number of A is *ReqA*: it is set in the Pi13. Request numbers on B are *ReqB* for reception and *FwdB* for transmission. Request number on C is *ReqC*.

After data processing, the Application submits a p1b8pe2e request to C, TYP=E, for request *ReqC*, to acknowledge the file or message received. This is a new request with number *ErpC*. Connect:Express prepares the PeSIT Message fpdu from the RENC file record that is accessed with the request number *ReqC* to retrieve original information.

B receives the EERP message for symbolic file *PI12*, from partner C, and saves it in the RENC file, with request number *ErpB*. C changes *ReqC* status to X, and B retrieves *FwdB* from information (Pi13+A+C+sent from A to C) and changes status to X. Partner A and C are configured in B with DPCSID ALIAS = **FWRD**. B knows that this is a end to end message (from the Pi11). B launches UEXFWRD at end of reception, using p1b8pe2e with FUN=F, type of request = E. All end to end transfer PeSIT fields are forwarded in the new transfer. A receives the EERP and saves it in the RENC file. A retrieves *ReqA* from information (Pi13+A+C+sent from A to C) and changes the status to X. B retrieves *ReqB* from information (Pi13+A+C+received from A to C) and changes the status to X.



Note : the difference between UEXROUT and UEXFWRD is that UEXROUT doesn’t transmit all the parameters . The forward request is a new request, with a new transfer identification (Pi13).

DN Control

This section provides information on what is DN Control and how to configure it in Connect:Express unix.

Overview

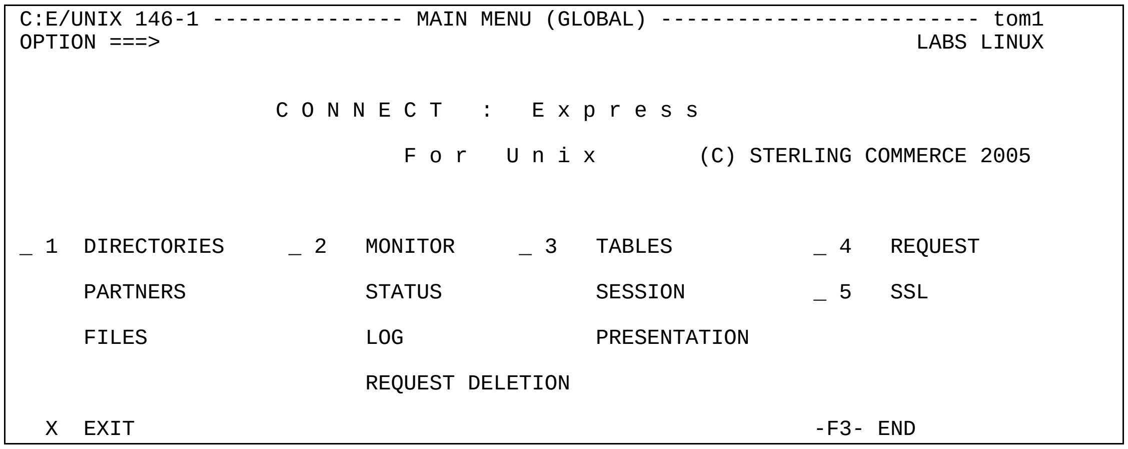
When receiving a certificate from a remote partner, the ssl interface (openssl) controls the validity of the certificate, and that the certificate is recorded in Connect:Express database. To control that the name in the certificate is what you expect it to be, you have to use the Certificate control function.

You can define the domain name, the DN, for an outbound session (Client Dn) and an inbound session (Server DN). For Connect:Express ssl server, the DN control is linked to the VERIFY OPTIONS of the SSL profile. The REMOTE CLIENT DN can be controlled only if the VERIFY OPTION is not 0, because this means that the Connect:Express server will receive a certificate from the Client. For Connect:Express ssl client, the DN control is done even if the VERIFY OPTION is not 2.

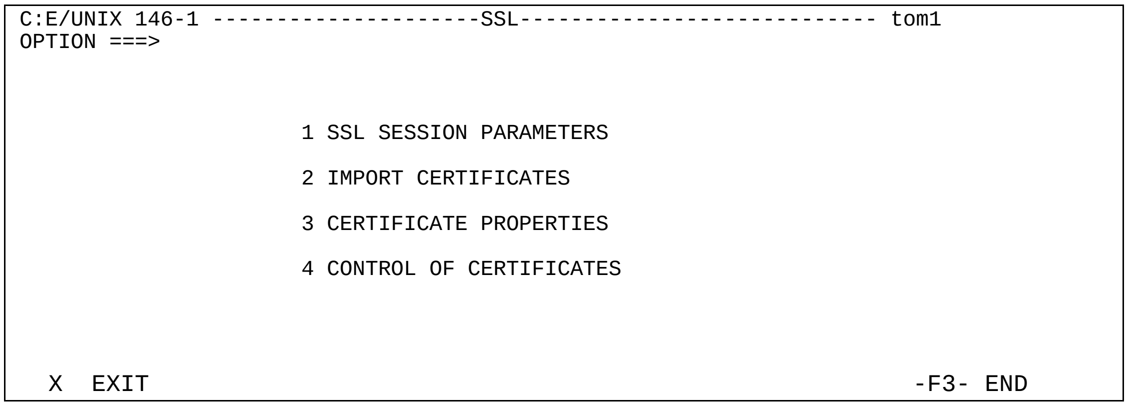
Operational Characteristics

DN control parameters are managed through STERM. You have to provide DN control definitions through the SSL menu, and to associate an existing DN control profile to a partner.

The SSL sub-menu shows option 4 CONTROL OF CERTIFICATES



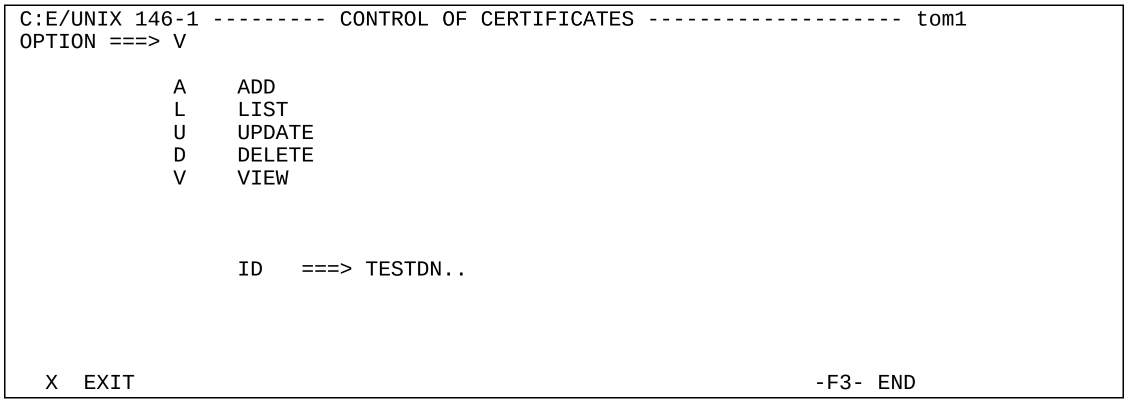
```
C:E/UNIX 146-1 --------------- MAIN MENU (GLOBAL) ------------------------- tom1
OPTION ===>                                                           LABS LINUX


                    C O N N E C T   :   E x p r e s s

                              F o r   U n i x       (C) STERLING COMMERCE 2005



_ 1  DIRECTORIES     _ 2   MONITOR     _ 3   TABLES          _ 4   REQUEST

     PARTNERS              STATUS            SESSION         _ 5   SSL

     FILES                 LOG               PRESENTATION

                           REQUEST DELETION

  X  EXIT                                                    -F3- END
```



```
C:E/UNIX 146-1 --------------------SSL---------------------------- tom1
OPTION ===>



                      1 SSL SESSION PARAMETERS

                      2 IMPORT CERTIFICATES

                      3 CERTIFICATE PROPERTIES

                      4 CONTROL OF CERTIFICATES




  X  EXIT                                                     -F3- END
```

A DN definition consists of one object DN and one issuer DN. The object DN is the name of the partner's certificate. The issuer DN is the name of the authority's certificate.

The figures below show DN definitions identified by the symbolic name TESTDN.



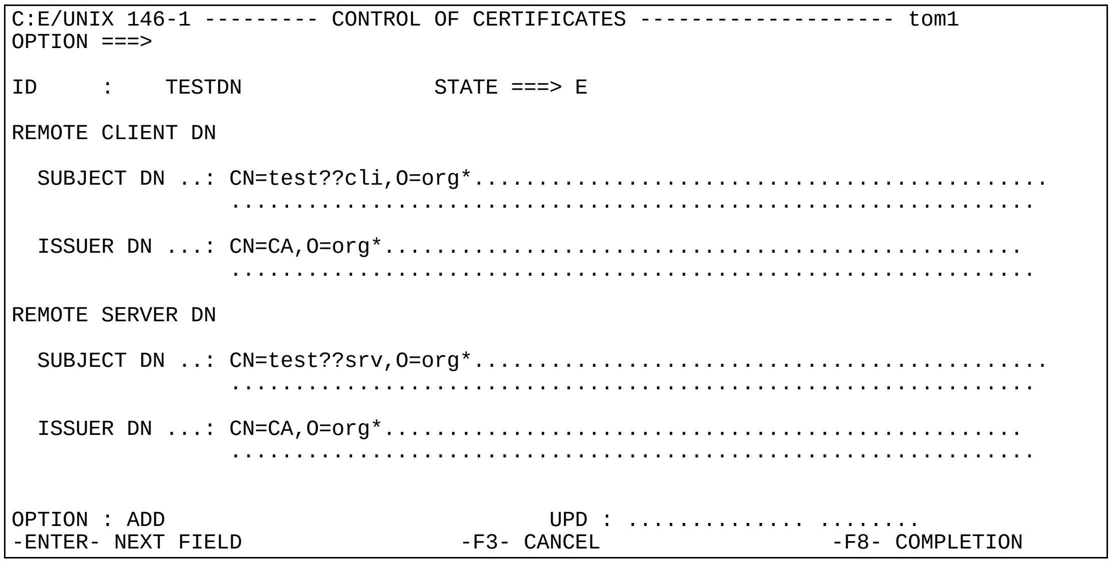
```
C:E/UNIX 146-1 --------- CONTROL OF CERTIFICATES -------------------- tom1
OPTION ===> V

            A    ADD
            L    LIST
            U    UPDATE
            D    DELETE
            V    VIEW



                 ID   ===> TESTDN..




  X  EXIT                                                      -F3- END
```

User can list existing definitions, add a definition, update, delete, view a definition.

Defining Certificates for Control

A Certificate control definition can contain both remote server and remote client DN definitions. During an outbound ssl session, the certificate of the server can be controlled against the REMOTE SERVER definition. During an inbound session, the certificate of the client can be controlled against the REMOTE CLIENT definition. Wild characters supported are '*' and '?'. They are used the same way as for file names.

The STATE field indicates if the control is active or not.



```
C:E/UNIX 146-1 --------- CONTROL OF CERTIFICATES -------------------- tom1
OPTION ===>

ID     :    TESTDN               STATE ===> E

REMOTE CLIENT DN

  SUBJECT DN ..: CN=test??cli,O=org*...........................................
                 ..............................................................

  ISSUER DN ...: CN=CA,O=org*................................................
                 ..............................................................

REMOTE SERVER DN

  SUBJECT DN ..: CN=test??srv,O=org*...........................................
                 ..............................................................

  ISSUER DN ...: CN=CA,O=org*................................................
                 ..............................................................


OPTION : ADD                              UPD : .............. ........
-ENTER- NEXT FIELD                 -F3- CANCEL                  -F8- COMPLETION
```

VIEW, UPDATE, DELETE: the date of last update is shown at the bottom.

```
C:E/UNIX 146-1 --------- CONTROL OF CERTIFICATES ------------------- tom1
OPTION ===>

ID     :    TESTDN               STATUS       :

REMOTE CLIENT DN

  SUBJECT DN ..: cn=test*cli,o=org*.............................................
                 ...............................................................

  ISSUER DN ...: cn=CA*,o=org*..................................................
                 ...............................................................

REMOTE SERVER DN

  SUBJECT DN ..: cn=test*srv,o=org*.............................................
                 ...............................................................

  ISSUER DN ...: cn=CA*,o=org*..................................................
                 ...............................................................


OPTION : UPDATE                           UPD : 10/01/18 11:41 username
-ENTER- NEXT FIELD                 -F3- CANCEL                  -F8- COMPLETION
```

The LIST shows DN control definitions, first part.

```
C:E/UNIX 146-1 --------- CONTROL OF CERTIFICATES ------------------- tom1
OPTION ===>                 REMOTE CLIENT DN

    ID       ST  SUBJECT DN                   ISSUER DN

    TESTDN   E   cn=test?cli,o=org*          cn=CA,o=org*
    TESTDN1  E   cn=testcli1,o=org*          cn=CA?,o=org*
    TESTDN2  E   cn=testcli2,o=org*          cn=CA,o=org*
    TESTDN3  E   cn=test*cli3,o=org*         cn=CA,o=org*





    <- -F10-  -F3- END  -F7- PREVIOUS SCREEN  -F8- NEXT SCREEN    -F11- ->
```

F10 or F11 display the second part

```
C:E/UNIX 146-1 --------- CONTROL OF CERTIFICATES ------------------- tom1
OPTION ===>                  REMOTE SERVER DN

    ID       ST  SUBJECT DN                   ISSUER DN

 -  TESTDN   E   cn=test?srv,o=org*           cn=CA,o=org*
 -  TESTDN1  E   cn=testsrv1,o=org*           cn=CA?,o=org*
 -  TESTDN2  E   cn=testsrv2,o=org*           cn=CA,o=org*
 -  TESTDN3  E   cn=testsrv3,o=org*           cn=CA,o=org*





    <- -F10-  -F3- END  -F7- PREVIOUS SCREEN  -F8- NEXT SCREEN    -F11- ->
```

Controlling Certificates of a Partner

A partner can be associated inbound and outbound DN definitions through a Certificate control entry. The partner definition is added the Certificate control entry field.

```
C:E/UNIX 146-1 --------------- PARTNERS DIRECTORY ------------------------- tom1
OPTION ===>

SYMBOLIC NAME        :    BOUCLE
PASSWORD ........... :    PSW            PASSWORD OF PARTNER
INITIALIZATION STATUS . : E              E:ENABLE     H:DISABLE
PARTNER TYPE .......... : O              T/O
PROTOCOL NUMBER ....... : 3              1:ETEBAC 3, 2:FTP, 3:PESIT
SESSION TABLE NUMBER .. : 1              1->9 SESSION TABLES
X25 PORT .............. :                X25 DEVICE NAME
MAX. NO. CONNECTIONS .. : 06/03/03       01->64  TOT/IN/OUT
TYPE OF CONNECTION .... : T              X, P, T OR M
X25 DIAL NUMBER ....... :                1-15 CHARACTERS
LOCAL DIAL NUMBER ..... :                1-15 CHARACTERS
EXTRA NETWORK FIELD ... :                'USER-DATA-FIELD'
FACILITIES ............ :
TCPIP HOST ............ : localhost                             PORT: 05015
TCPIP ADDRESS ......... :                DEF FTP FILE .. :
DPCSID ALIAS .......... : BOUCLE         SSLPARM ID .... :
DPCPSW ALIAS .......... : PSW            CERTIFICATE CONTROL : TESTDN
NUMBER OF RETRIES ..... :                INTERV.SESS,TRF ... :   ,   MINUTES
DO YOU WANT TO GO ON ?
OPTION : VIEW                            UPD : 10/02/26 11:53 gcz
-ENTER- NEXT FIELD               -F3- CANCEL                 -F8- COMPLETION
```

Return codes are:

Inbound	: prc=302 partner inbound unauthorized
Outbound	: prc=304 identification of called partner unauthorized

trc=2057 subject DN don't match
trc=2058 issuer DN don't match
trc=2059 DN record not found

FTP Extended Identification

FTP Extended identification enables the use of long FTP user names and passwords, in place of the symbolic 8 Characters values, and enables communications through firewalls.

For example, connecting to an FTP server through a firewall is done with this syntax:

```
        USER my_username_any_length@server_host_name_any_length
        PASSWORD my_password_any_length
```

The FTP Extended Identification File

You can define extended FTP identifications in the apmftpe file placed in the config directory:

```
$TOM_DIR/config/apmftpe
```

You can create and update this file through an editor, using the syntax shown below: Keyword ID is required, commas are used as separators. Extended id or extended password can be omitted.

```
               ID=Aliasid,Ftp_Extended_Id,Ftp_Extended_Password
         or
               ID=Aliasid,Ftp_Extended_Id              (no password)
         or
               ID=Aliasid,,Ftp_Extended_Password       (no user name)
```

The FTP DPCSID ALIAS Field

The partner definition provides the DPCSID ALIAS field. If this field is set with $$FTPE$$ keyword, the alias id must be picked up in the apmftpe file, identified by the partner name. if the DPCSID ALIAS field is set with a name starting with '$$', $$*alias* for example, the alias id must be picked up in the apmftpe file, identified by the name "alias" from $$*alias*.

```
C:E/UNIX 146-1 --------------- PARTNERS DIRECTORY ------------------------- tom1
OPTION ===>

SYMBOLIC NAME ......... : PFTP01
PASSWORD .............. : PASSWD          PASSWORD OF PARTNER
INITIALIZATION STATUS . : E               E:ENABLE     H:DISABLE
PARTNER TYPE .......... : O               T/O
PROTOCOL NUMBER ....... : 2               1:ETEBAC 3, 2:FTP, 3:PESIT
SESSION TABLE NUMBER .. : 1               1->9 SESSION TABLES
X25 PORT .............. :                 X25 DEVICE NAME
MAX. NO. CONNECTIONS .. : 20/10/10        01->64  TOT/IN/OUT
TYPE OF CONNECTION .... : T               X, P, T OR M
X25 DIAL NUMBER ....... :                 1-15 CHARACTERS
LOCAL DIAL NUMBER ..... :                 1-15 CHARACTERS
EXTRA NETWORK FIELD ... :                 'USER-DATA-FIELD'
FACILITIES ............ :
TCPIP HOST ............ : localhost                         PORT . : 05050
TCPIP ADDRESS ......... :                 DEF FTP FILE .. : FFTP01
DPCSID ALIAS .......... : $$FTPE$$        SSLPARM ID .... :
DPCPSW ALIAS .......... : PASSWD          CONTROL OF CERTIFICATES  :
NUMBER OF RETRIES ..... :                 INTERV.SESS ,TRF  :   ,    MINUTES
DO YOU WANT TO GO ON ?
OPTION : VIEW                             UPD : 10/04/01 15:33 gcz
-ENTER- NEXT FIELD                 -F3- CANCEL                 -F8- COMPLETION
```

For PFTP01, the apmftpe file will provide the ID=PFTP01 information.

```
               ID=PFTP01,Ftp_Extended_pftp01,Ftp_pftp01_Extended_Password
               ID=alias,my_alias,my_alias_Password
               ID=SPECID,my_user@address_of_server_destination.com
```

You can pass the DPCSID ALIAS with the transfer request, from sterm or itom. For example:

```
$TOM_DIR/itom/p1b8preq "/SFN=FILE01/DIR=T/SPN=PFTP01/SID=$$SPECID"
                       "/DSN=\$TOM_DIR/config/sysin"
```

For this request, the apmftpe file will provide the ID=SPECID information.

If a field is omitted in the apmftpe file for the current definition, the value is taken from the directory, or the default sysin value. For example SPECID does not have a password: the password is PASSWD from the directory DPCPSW ALIAS field. If no extended user id is provided, the DPCSID name from the sysin is used: this is the way to just providing an extended password.

The End to End Utility P1b8pe2e

The end to end utility, called p1b8pe2e, enables you to forward and acknowledge transfers of files and messages.

Acknowledging a Transfer

If the request is present in the RENC file, it is possible to acknowledge it by referencing its number.

```
$TOM_DIR/itom/p1b8pe2e   "/FUN=E/REQ=10400065/SPN=adjacent"  "/ACK='feedback message'"
```

The SPN parameter is necessary if the initial node is not the adjacent partner.

If the request is no longer in the RENC file, all parameters from the initial transfer must be provided.

```
$TOM_DIR/itom/p1b8pe2e "/FUN=E/SPN=adjacent"
                       "/P12=filef/P11=XX/P03=oo/P04=dd/P13=id/p51=dh/p61=cc/p62=bb"
                       "/ACK='feedback message'"
```

Forwarding a Transfer

If the request is present in the RENC file, it is possible to forward it by referencing its number.

```
$TOM_DIR/itom/p1b8pe2e   "/FUN=F/REQ=10400065/SPN=adjacent"
```

The SPN parameter is required.

If the request is no longer in the RENC file, all parameters from the initial transfer must be provided.

```
$TOM_DIR/itom/p1b8pe2e "/FUN=F,TYP=N/SPN=adjacent"
                       "/P12=file/P11=XX/P03=oo/P04=dd/P13=id/p51=dh/p61=cc/p62=bb"
```

P1b8pe2e Reference

This section provides the syntax rules and all parameters that apply to p1b8pe2e utility.

P1b8pe2e utility can receive one to five parameters, depending on the type of function used and the way the transfer definition is passed. Parameter #1 can provide general transfer request parameters such as priority, notification options, link, scheduling date etc ….

The tables below list the parameters and sub-parameters and provide a description and rules for each.

EERP - Request

This request refers to the reception initial request, using the /REQ= subparameter.

Parameter	Subparameter	Description	Required/Default
#1	/FUN	Function E=EERP	Yes
	/REQ	Request number, 8 alphanumeric characters. Example: /REQ=09800005	Yes
	/SPN	Remote partner name (adjacent)	Yes
	/SID	Local name (alias)	RPAR/sysin
	/PSW	Local password	RPAR/sysin
	/NTF	Notification options	RFIC
	/PRT	Priority	RFIC
	/LNK	Link type	RPAR
	/DAT	Scheduling date	Immediate
	/FAG	File Agent option	'N'
#2 /ACK /DSN		Eerp acknowledgment (message or file) default from the $$EERP$$ definition. Eerp acknowledgment (message) Eerp acknowledgment (file)	$$EERP$$

EERP - Transfer Definition

This request provides the initial request information. No /REQ= parameter is provided, all transfer information is provided in parameter #2.

Parameter	Subparameter	Description	Required/Default
#1	/FUN	Function E=EERP	Yes
	/SPN	Remote partner name	Yes
	/SID	Local name (alias)	RPAR/sysin
	/PSW	Local password	RPAR/sysin
	/NTF	Notification options	RFIC
	/PRT	Priority	RFIC
	/LNK	Link type	RPAR
	/DAT	Scheduling date	Immediate
	/FAG	File Agent option	'N'
#2		Transfer definition	Yes
	/ORG	Origin of transfer. 1 to 8 alphanumeric characters. (pi3) Example: /ORG=Orgtrf01	Yes
	/DST	Destination of transfer. 1 to 8 alphanumeric characters. (pi4) Example: /DST=DSTtrf01	Yes
	/P11	File type. 4 hexadecimal characters. (pi11) Example: /P11=01FA	Yes
	/P12	File name. 1 to 8 alphanumeric characters. (pi12) – RFIC definition. Example: /P12=Ftest01	Yes
	/P13	Transfer identification. 1 to 8 numeric characters. (pi13) Example /P13=18	Yes
	/P51	File creation date: 12 numeric characters. Example: /P51=040110092503	Yes
	/P61	Transfer sender: 0 to 24 characters. (pi61) Example: /P61=Client name	Yes

	/P62	Transfer receiver: 0 to 24 characters. (pi62) Example: /P62=Service name	Yes
#3 /ACK /DSN		Eerp acknowledgment (message or file) default from the $$EERP$$ definition. Eerp acknowledgment (message) Eerp acknowledgment (file)	$$EERP$$

Forwarding a Request

This request refers to the reception initial request. Only parameter #1 is provided. /DSN, /P99, /LAB are invalid as these information are retreived in the RENC information for the initial request.

Parameter	**Subparameter**	**Description**	**Required/Default**
#1	/FUN	Function F=Forward	Yes
	/REQ	Request number, 8 alphanumeric characters. Example: /REQ=09800005	Yes
	/SPN	Remote partner name	Yes
	/SID	Local name (alias)	RPAR/sysin
	/PSW	Local password	RPAR/sysin
	/NTF	Notification options	RFIC
	/PRT	Priority	RFIC
	/LNK	Link type	RPAR
	/DAT	Scheduling date	Immediat
	/FAG	File Agent option	‘N’

Forwarding a Transfer Definition

This request provides the initial request information. No /REQ= parameter is provided.

Parameter	**Subparameter**	**Description**	**Required/Default**
#1	/FUN	Function F=Forward	Yes
	/SPN	Remote partner name	Yes
	/SID	Local name (alias)	RPAR/sysin
	/PSW	Local password	RPAR/sysin
	/NTF	Notification options	RFIC
	/PRT	Priority	RFIC
	/LNK	Link type	RPAR
	/DAT	Scheduling date	Immediat
	/FAG	File Agent option	‘N’
#2		Transfer definition	Yes
	/ORG	Origine of transfer. 1 to 8 alphanumeric characters. (pi3) Example: /ORG=Orgtrf01	Yes
	/DST	Destination of transfer. 1 to 8 alphanumeric characters. (pi4) Example: /DST=DSTtrf01	Yes
	/P11	File type. 4 hexadecimal characters. (pi11) Example: /P11=01FA	Yes
	/P12	File name. 1 to 8 alphanumeric characters. (pi12) – RFIC definition. Example: /P12=Ftest01	Yes
	/P13	Transfer identification. 1 to 8 numeric characters. (pi13) Example /P13=18	Yes
	/P51	File creation date: 12 numeric characters. Example: /P51=040110092503	Yes
	/P61	Transfer sender: 0 to 24 characters. (pi61)	Yes

		Example: /P61=Client name	
	/P62	Transfer receiver: 0 to 24 characters. (pi62) Example: /P62=Service name	Yes
#3 #4 #5 /DSN /P99 /LAB		 Physical file name User data File label	 RFIC RFIC

Error Codes

This section provides the meaning of the return code from p1b8pe2e utility. The return code is a 4 characters field structured as shown below.

Field	**Definition**
1	1 numeric character: parameter value – from 1 to 5
2	2 numeric characters sub parameter value. 00 Other 01 Priority 02 Direction 03 Link 04 Partner 05 File 06 Physical Name 07 User Data Field 08 Date 09 Monitor 10 Request Number 11 Alias Name 12 Alias Password 13 Record Format 14 Record Length 15 Api 16 State 17 Request Type 18 Type/Struct/Mode FTP 19 Store/Unique FTP 20 File agent flag Y/N 21 Label 22 Pi99 254 23 User Origin 24 User Destination 25 Pi61 26 Pi62 27 Julian Date 28 Notification 29 Eerp/snf pi11 30 Eerp/snf pi12 31 Eerp/snf pi13 32 Eerp/snf pi51 33 Eerp ACK 34 Eerp or FWD
3	1 numeric character: error code: 1 Invalid Field 2 Duplicate Field 3 Invalid Field Length 4 Missing Required Field

Example: 2331 is for parameter 2, subparameter pi13, invalid length.

New Variables, Parameters and Codes

The latest modifications and new functionalities have introduced changes in Connect:Express general fields.

Symbolic variable

&IDT variable contains the decimal value of Pi13, the transfer id, that goes through the forward process forth and back. This variable is processed in received dsn, and in label and p99 both directions

User Command Parameters

All parameters required for store and forward process are now available, in order to enable automatic procedures. The request type is added to make the difference between file, message and EERP.

1. Request number of transfer
2. Symbolic file name
3. Symbolic partner name
4. Physical file name (absolute path)
5. Direction of transfer (R,T)
6. System Return Code (SRC)
7. Connect:Express Return Code (TRC)
8. Protocol Return Code (PRC)
9. Received Pi99
10. Sent Pi99
11. Transfer Origin
12. Transfer Destination
13. Local Name
14. Label
15. User sender (Pi61)
16. User receiver (Pi62)
17. Request start date
18. Request start time
19. Transfer status
20. Julian date
21. Number of records
22. Number of bytes
23. Request end date
24. Request end time
25. Type of request (N,I,H,M,E)
26. File type (4 hexadecimal characters)
27. Transfer ID (8 decimal characters)
28. File date-time (12 characters)

If parameter **5.** (Transfer direction) is ‘R’ and parameter **25.** (Type of request) is ‘N’ or ‘M’, the file or message can be acknowledged using parameters **2.** , **3.** , **11.**, **12.** , **15.** , **16.** , **26.** , **27.** , **28.** .
If parameter **5.** (Transfer direction) is ‘R’ and parameter **20.** (Type of request) is ‘N’ or ‘M’ or ‘E’, the file, message or EERP can be forwarded, same Type of request **20.** , using parameters **2.** , **3.** , **11.**, **12.** , **15.** , **16.** , **26.** , **27.** , **28.**.

TRC codes

2050	EERP or a forward is requested for a request that is not yet ended
2051	EERP or a forward is requested for a transmission request , received for a reception request
2053	EERP is requested/received for a previous EERP
2055	EERP or a forward is requested for a request that has already been acknowledged
2057	The partner’s subject dn is invalid
2058	The partner’s issuer dn is invalid
2059	The control dn record is not found
2219	The ftp extension definition is not found