Connecting UES-Enabled Databases

Overview

Prior to Adabas Version 7, Entire Net-Work converted all data for mainframe Adabas when necessary from ASCII to EBCDIC. Starting with Version 7, Adabas is delivered with its own data conversion capability called universal encoding support (UES). Entire Net-Work detects when it is connected to a target database that converts data and passes the data through to Adabas without converting it.

For Adabas Version 7.4, UES is enabled by default for the link routines ADALNK, ADALNKR, ADALCO, and ADALNI.

Note:
The use of UES-enabled link routines is transparent to applications, including applications that do not require UES translation support: it is not necessary to disable UES support.

Load Modules

The load modules for ADALNK, ADALNKR, and ADALCO have been linked with LNKUES and the default translation tables.

LNKUES converts data in the Adabas buffers and byte-swaps, if necessary, depending on the data architecture of the caller.

The two standard translation tables are

- ASC2EBC: ASCII to EBCDIC translation; and
- EBC2ASC: EBCDIC to ASCII translation.

The Adabas translation table pair is provided in the section *Translation Tables*.

Default or Customized Translation Tables

You may use the load modules with the default translation tables linked in, or you may prepare your own customized translation tables, re-assemble the tables, and link them with the LNKUES module that is delivered.

Notes:

1. It should only be necessary to modify these translation tables in the rare case that some country-specific character other than "A-Z a-z 0-9" must be used in the Additions 1 (user ID) or Additions 3 field of the control block.
2. The load module LNKUESL delivered with earlier levels of Adabas Version 7 is no longer supplied since the link jobs now specify the LNKUES module and the translation tables separately.
3. The LNKUES module is functionally reentrant; however, it is not linked that way in the Adabas load library.
4. When linking the LNKUES load module and the translation tables, the linkage editor may produce warning messages concerning the reentrant or reusability status of the linked module. These warning messages can be ignored.

Source Modules

The ADALNK, ADALNKR, ADALCO, and ADALNI source modules have been coded to enable UES support by default when assembled:

- The &UES Boolean assembly variable is set to 1 by default; the statement to set it to 0 has been commented out.
- The setting of the other Boolean variables and equates such as the SVC number and the database ID remain unchanged from earlier deliveries of the source modules.

Required Environment

The Adabas database must be UES-enabled. See *DBA Tasks* and the ADACMP and ADADEF utility sections in the *Adabas Utilities* documentation for more information.

Connection Possibilities

UES-enabled databases are connected to machines with different architectures through Smarts, through Entire Net-Work, and optionally in a z/OS environment, through a direct TCP/IP link to the Adabas nucleus from web-based applications or from PC-based applications such as Software AG’s Jadabas.

Connection Through Com-plete or Smarts

Adabas SQL Gateway (ACE) clients may not be strictly EBCDIC in an environment where databases are connected through Software AG’s internal product Smarts (APS).

The relevant Adabas link routine ADALCO is UES-enabled by default. The sample jobstream to assemble and link the ADALCO module is ALNKLCO.

The assembled and linked ADALCO (as delivered or with customized and reassembled translation tables) is placed in the Smarts steplib or a user library concatenated with it.

- Step 1: Assemble ADALCO Module into the Adabas Load Library (SMA Job Number I070)
- Step 2: Assemble Translation Tables into the Adabas Load Library (SMA Job Number I056)
- Step 3: Link the Translation Tables and LNKUES into ADALCO (SMA Job Number I088)
- Step 4: Make ADALCO Available to Smarts

Step 1: Assemble ADALCO Module into the Adabas Load Library (SMA Job Number I070)

Modify the MVSJOBS member ALNKLCO to assemble and link ADALCO as follows:

- provide all necessary jobcard information
- check the symbolic parameter value for version, revision level, and SM level (*vrs*). It must reflect the level of your Adabas source and load libraries.
- check the data set names for SYSLIB, SYSIN, SYSLMOD & SYSLIN in the SAGASM and LINKALL inline procedures.

```
// JOB
//SAGASM PROC MEM=,
// VRS=
//ASM EXEC PGM=ASMA90,
// PARM='ASA,NODECK,OBJECT,USING(MAP),XREF(SHORT),TERM'
//SYSUT1 DD SPACE=(4096,(120,120),,,ROUND),UNIT=VIO,DCB=BUFNO=1
//SYSTERM DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSLIB DD DISP=SHR,DSN=ADABAS.&VRS..SRCE
// DD DISP=SHR,DSN=SYS1.MACLIB
// DD DISP=SHR,DSN=SYS1.MODGEN
//SYSIN DD DISP=SHR,DSN=ADABAS.&VRS..SRCE(&MEM)
//SYSLIN DD DSN=&&OBJ,SPACE=(3040,(40,40),,,ROUND),UNIT=VIO,
// DISP=(MOD,PASS),
// DCB=(BLKSIZE=3040,LRECL=80,RECFM=FBS,BUFNO=1)
//LINK EXEC PGM=HEWL,REGION=2M,COND=(5,LT,ASM),
// PARM='XREF,LIST(ALL),LET,MAP'
//SYSPRINT DD SYSOUT=*
//SYSLIN DD DSN=&&OBJ,DISP=(OLD,DELETE)
//SYSLMOD DD DISP=SHR,DSN=ADABAS.&VRS..LOAD(&MEM)
// PEND
//*
//ADALCO EXEC SAGASM,VRS=Vvrs,MEM=ADALCO
//LINK.SYSIN DD *
MODE AMODE(31) RMODE(ANY)
ENTRY ADABAS
NAME ADALCO(R)
```

Step 2: Assemble Translation Tables into the Adabas Load Library (SMA Job Number I056)

Assemble the ASCII to EBCDIC and EBCDIC to ASCII translation tables, either default or customized.

```
/*
//ASC2EBC EXEC SAGASM,VRS=Vvrs,MEM=ASC2EBC
//LINK.SYSIN DD *
MODE AMODE(31) RMODE(ANY)
ENTRY ASC2EBC
NAME ASC2EBC(R)
/*
//EBC2ASC EXEC SAGASM,VRS=Vvrs,MEM=EBC2ASC
//LINK.SYSIN DD *
MODE AMODE(31) RMODE(ANY)
ENTRY EBC2ASC
NAME EBC2ASC(R)
```

Step 3: Link the Translation Tables and LNKUES into ADALCO (SMA Job Number I088)

Link the ADALCO, ASC2EBC, EBC2ASC and LNKUES modules into a final ADALCO module that is UES-enabled. Place this load module into a “USER.LOAD” library. Be sure to modify the &USERLIB symbol in the SYSLMOD statement to match your user load library.

```
/*
//LINKALL PROC VRS=,USERLIB=
//LKED EXEC PGM=HEWL,REGION=2M,COND=(5,LT),
// PARM='XREF,LIST(ALL),LET,MAP,NCAL'
//SYSPRINT DD SYSOUT=*
//ADALIB DD DISP=SHR,DSN=ADABAS.&VRS..LOAD
//SYSLMOD DD DISP=SHR,DSN=&USERLIB
//SYSLIN DD DDNAME=SYSIN
// PEND
//LINKUES EXEC LINKALL,VRS=Vvrs,
// USERLIB='YOUR.USER.LOADLIB'

//LKED.SYSIN DD *
MODE AMODE(31) RMODE(ANY)
INCLUDE ADALIB(ADALCO)
INCLUDE ADALIB(LNKUES)
INCLUDE ADALIB(ASC2EBC)
INCLUDE ADALIB(EBC2ASC)
ENTRY ADABAS
NAME ADALCO(R)
/*
```

Step 4: Make ADALCO Available to Smarts

The (re)linked ADALCO must be made available to Smarts. If you are calling Adabas Version 7 and you do not have the correct LNKUES/ADALCO module, Adabas produces unexpected results: response code 022, 253, etc.

Connection Through Entire Net-Work

UES-enabled databases are connected through Software AG's Entire Net-Work (WCP) using the Adabas non-reentrant batch or TSO link routine ADALNK. The sample jobstream to assemble and link the non-reentrant ADALNK module is ALNKLNK.

The assembled and linked non-reentrant, batch ADALNK (as delivered or with customized and reassembled translation tables) is placed in the Entire Net-Work steplib.

- Step 1: Assemble ADALNK Module into Adabas Load Library (SMA Job Number I055)
- Step 2: Assemble Translation Tables into Adabas Load Library (SMA Job Number I056)
- Step 3: Link the Translation Tables and LNKUES into ADALNK (SMA Job Number I088)
- Step 4: Make ADALNK Available to Entire Net-Work

Step 1: Assemble ADALNK Module into Adabas Load Library (SMA Job Number I055)

Modify the MVSJOBS member ALNKLNK to assemble and link ADALNK as follows:

- provide all necessary jobcard information
- check the symbolic parameter value for version, revision level, and SM level (*vrs*). It must reflect the level of your Adabas source and load libraries.
- check the data set names for SYSLIB, SYSIN, SYSLMOD, and SYSLIN in the SAGASM and LINKALL inline procedures.

```
// JOB
//SAGASM PROC MEM=,
// VRS=
//ASM EXEC PGM=ASMA90,
// PARM='ASA,NODECK,OBJECT,USING(MAP),XREF(SHORT),TERM'
//SYSUT1 DD SPACE=(4096,(120,120),,,ROUND),UNIT=VIO,DCB=BUFNO=1
//SYSTERM DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSLIB DD DISP=SHR,DSN=ADABAS.&VRS..SRCE
// DD DISP=SHR,DSN=SYS1.MACLIB
// DD DISP=SHR,DSN=SYS1.MODGEN
//SYSIN DD DISP=SHR,DSN=ADABAS.&VRS..SRCE(&MEM)
//SYSLIN DD DSN=&&OBJ,SPACE=(3040,(40,40),,,ROUND),UNIT=VIO,
// DISP=(MOD,PASS),
// DCB=(BLKSIZE=3040,LRECL=80,RECFM=FBS,BUFNO=1)
//LINK EXEC PGM=HEWL,REGION=2M,COND=(5,LT,ASM),
// PARM='XREF,LIST(ALL),LET,MAP'
//SYSPRINT DD SYSOUT=*
//SYSLIN DD DSN=&&OBJ,DISP=(OLD,DELETE)
//SYSLMOD DD DISP=SHR,DSN=ADABAS.&VRS..LOAD(&MEM)
// PEND
//*
//ADALNK EXEC SAGASM,VRS=Vvrs,MEM=ADALNK
//LINK.SYSIN DD *
MODE AMODE(31) RMODE(24)
ENTRY ADABAS
NAME ADALNK(R)
```

Step 2: Assemble Translation Tables into Adabas Load Library (SMA Job Number I056)

If you prefer to use the same translation tables that are used in Entire Net-work, change the COPY statements in ASC2EBC and EBC2ASC from UES2ASC and UES2EBC to NW2ASC and NW2EBC, respectively. After modifying the translation tables, be sure to (re)assemble them and link them with the delivered LNKUES module.

The Entire Net-Work translation table pair is also provided in the section *Translation Tables*.

Assemble the ASCII to EBCDIC and EBCDIC to ASCII translation tables, either default or customized.

```
/*
//ASC2EBC EXEC SAGASM,VRS=Vvrs,MEM=ASC2EBC
//LINK.SYSIN DD *
MODE AMODE(31) RMODE(ANY)
ENTRY ASC2EBC
NAME ASC2EBC(R)
/*
//EBC2ASC EXEC SAGASM,VRS=Vvrs,MEM=EBC2ASC
//LINK.SYSIN DD *
MODE AMODE(31) RMODE(ANY)
ENTRY EBC2ASC
```

Step 3: Link the Translation Tables and LNKUES into ADALNK (SMA Job Number I088)

Link the ADALNK, ASC2EBC, EBC2ASC, LNKUES, and other user exit modules into a final ADALNK module that is UES-enabled. Place this load module into a “USER.LOAD” library. Be sure to modify the &USERLIB symbol in the SYSLMOD statement to match your user load library.

```
/*
//LINKALL PROC VRS=,USERLIB=
//LKED EXEC PGM=HEWL,REGION=2M,COND=(5,LT),
// PARM='XREF,LIST(ALL),LET,MAP,NCAL'
//SYSPRINT DD SYSOUT=*
//ADALIB DD DISP=SHR,DSN=ADABAS.&VRS..LOAD
//SYSLMOD DD DISP=SHR,DSN=&USERLIB
//SYSLIN DD DDNAME=SYSIN
// PEND
//LINKUES EXEC LINKALL,VRS=Vvrs,
// USERLIB='YOUR.USER.LOADLIB'
//LKED.SYSIN DD *
MODE AMODE(31) RMODE(24)
INCLUDE ADALIB(ADALNK)
INCLUDE ADALIB(LNKUES)
INCLUDE ADALIB(ASC2EBC)
INCLUDE ADALIB(EBC2ASC)
ENTRY ADABAS
NAME ADALNK(R)
/*
```

Step 4: Make ADALNK Available to Entire Net-Work

The (re)linked ADALNK must be made available to Entire Net-Work. If you are calling Adabas Version 7 and you do not have the correct LNKUES/ADALNK module, Adabas produces unexpected results: response code 022, 253, etc.

Connection Through a Direct TCP/IP Link

A TCP/IP link requires in addition that you link a reentrant ADALNKR module with LNKUES and your customized and reassembled translation tables and that you make the result available in the Adabas steplib.

UES-enabled databases are connected directly through TCP/IP using the Adabas reentrant batch or TSO link routine ADALNKR. The sample jobstream to assemble and link the ADALNK module is ALNKLNKR.

- Step 1: Assemble the ADALNKR Module into the Adabas Load Library
- Step 2: Assemble the Two Translation Tables into the Adabas Load Library (SMA Job Number I056)
- Step 3: Link the Translation Tables and LNKUES into ADALNKR
- Step 4: Make ADALNKR Available to the Adabas Nucleus

Step 1: Assemble the ADALNKR Module into the Adabas Load Library

In order to enable UES support for a database through TCP/IP, you must prepare a modified batch ADALNKR.

to prepare a modified batch ADALNKR:

1. Update the source ADALNKR:

   ```
   &RENT SETB 1
   SVCNR EQU nnn (hard-coded SVC number)
   ```

2. Assemble and link the modified batch ADALNKR.

   Modify the MVSJOBS member ALNKLNKR to assemble and link ADALNKR as follows:

   - provide all necessary jobcard information
   - check the symbolic parameter value for version, revision level, and SM level (vrs). It must reflect the level of your Adabas source and load libraries.
   - check the data set names for SYSLIB, SYSIN, SYSLMOD, and SYSLIN in the SAGASM and LINKALL inline procedures.

     ```
     // JOB
     //SAGASM PROC MEM=,
     // VRS=
     //ASM EXEC PGM=ASMA90,
     // PARM='ASA,NODECK,OBJECT,USING(MAP),XREF(SHORT),TERM'
     //SYSUT1 DD SPACE=(4096,(120,120),,,ROUND),UNIT=VIO,DCB=BUFNO=1
     ```

```
        //SYSTERM DD SYSOUT=*
        //SYSPRINT DD SYSOUT=*
        //SYSLIB DD DISP=SHR,DSN=ADABAS.&VRS..SRCE
        // DD DISP=SHR,DSN=SYS1.MACLIB
        // DD DISP=SHR,DSN=SYS1.MODGEN
        //S//SYSLIN DD DSN=&&OBJ,SPACE=(3040,(40,40),,,ROUND),UNIT=VIO,
        // DISP=(MOD,PASS),
        // DCB=(BLKSIZE=3040,LRECL=80,RECFM=FBS,BUFNO=1)
        //LINK EXEC PGM=HEWL,REGION=2M,COND=(5,LT,ASM),
        // PARM='XREF,LIST(ALL),LET,MAP'
        //SYSPRINT DD SYSOUT=*
        //SYSLIN DD DSN=&&OBJ,DISP=(OLD,DELETE)
        //SYSLMOD DD DISP=SHR,DSN=ADABAS.&VRS..LOAD(&MEM)
        // PEND
        //*
        //ADALNKR EXEC SAGASM,VRS=Vvrs,MEM=ADALNKR
        //LINK.SYSIN DD *
        MODE AMODE(31) RMODE(24)
        ENTRY ADABAS
        NAME ADALNKR(R)YSIN DD DISP=SHR,DSN=ADABAS.&VRS..SRCE(&MEM)
```

Step 2: Assemble the Two Translation Tables into the Adabas Load Library (SMA Job Number I056)

Assemble the ASCII to EBCDIC and EBCDIC to ASCII translation tables, either default or customized.

```
/*
//ASC2EBC EXEC SAGASM,VRS=Vvrs,MEM=ASC2EBC
//LINK.SYSIN DD *
MODE AMODE(31) RMODE(ANY)
ENTRY ASC2EBC
NAME ASC2EBC(R)
/*
//EBC2ASC EXEC SAGASM,VRS=Vvrs,MEM=EBC2ASC
//LINK.SYSIN DD *
MODE AMODE(31) RMODE(ANY)
ENTRY EBC2ASC
NAME EBC2ASC(R)
```

Step 3: Link the Translation Tables and LNKUES into ADALNKR

It is now necessary to (re)link ADALNKR with LNKUES and your customized and reassembled translation tables.

Link the ADALNKR, ASC2EBC, EBC2ASC, LNKUES, and other user exit modules into a final ADALNKR module that is UES-enabled. Place this load module into a “USER.LOAD” library. Be sure to modify the &USERLIB symbol in the SYSLMOD statement to match your user load library.

```
/*
//LINKALL PROC VRS=,USERLIB=
//LKED EXEC PGM=HEWL,REGION=2M,COND=(5,LT),
// PARM='XREF,LIST(ALL),LET,MAP,NCAL'
//SYSPRINT DD SYSOUT=*
//ADALIB DD DISP=SHR,DSN=ADABAS.&VRS..LOAD
//SYSLMOD DD DISP=SHR,DSN=&USERLIB
//SYSLIN DD DDNAME=SYSIN
// PEND
//LINKUES EXEC LINKALL,VRS=Vvrs,
// USERLIB='YOUR.USER.LOADLIB'
```

```
//LKED.SYSIN DD *
MODE AMODE(31) RMODE(24)
INCLUDE ADALIB(ADALNKR)
INCLUDE ADALIB(LNKUES)
INCLUDE ADALIB(ASC2EBC)
INCLUDE ADALIB(EBC2ASC)
ENTRY ADABAS
NAME ADALNKR(R)
/*
```

Step 4: Make ADALNKR Available to the Adabas Nucleus

The (re)linked ADALNK must be made available to the Adabas nucleus.

If you are calling Adabas Version 7 directly through a TCP/IP link and the correct ADALNKR is not available to the Adabas nucleus, Adabas produces unexpected results: response code 148, empty buffers, etc.

Activating the TCP/IP Link

To activate a direct TCP/IP link to the Adabas nucleus:

1. Set the ADARUN parameter TCPIP=YES.
2. Specify a universal resource locator (URL).

Specifying a URL

The URL is a 20-byte address that conforms to the RFC specification for URLs.

You can specify the URL required to activate the direct TCP/IP link in the ADARUN parameter TCPURL as follows:

```
ADARUN PROG=ADANUC,TCPIP=YES,TCPURL=api-name://stackid:port-number

—where

api-name is a 1-3 character value identifying the application programming interface (API) to use.
The APIs for the IBM TCP/IP stack (HPS, OES) are currently supported.
stackid is a 1-8 character value identifying the stack to use:
- for the HPS API, this is the name of the TCP/IP started task.
- for the OES API, no value is needed.
- for the ILK API, this is the subsystem identifier.
port-number is a 1-5 character number in decimal notation.
```

Examples

```
ADARUN PROG=ADANUC,TCPIP=YES,TCPURL=HPS://STACKNAME:1234
ADARUN PROG=ADANUC,TCPIP=YES,TCPURL=OES://:1234
ADARUN PROG=ADANUC,TCPIP=YES,TCPURL=ILK://ILZ5:1234
```

Managing URLs

Optionally, you can specify the first and additional URLs using the operator command TCPIP:

```
TCPIP={ OPEN=url|CLOSE=url | CLOSE }
```

—where *url* is the URL for the TCP/IP link you want to open or close and has the same format as the ADARUN TCPURL parameter:

```
api-name://stackid:port-number
```

The command allows you to open or close a TCP/IP link to the Adabas nucleus or to close all links. It can only be used when ADARUN TCPIP=YES and all conditions for that setting have been met. This command can be used to close the URL set in the ADARUN TCPURL parameter, or to open/close additional TCP/IP links.

Examples

```
TCPIP=OPEN=ILK://ILZ5:1234
TCPIP=CLOSE=ILK://ILZ5:1234

To close all open URLs:
TCPIP=CLOSE
```