EntireX Broker ACI for PL/I

Advanced communication interface. ACI-based programming is the base technology of EntireX. It uses a traditional Application Programming Interface (API) approach for conducting client/server and peer-to-peer dialog between distributed processes.

This chapter describes the EntireX Broker ACI from the perspective of the programming language PL/I and covers the following topics:

- Call Format
- Broker ACI Control Block Layout
- Broker ACI Control Block Copybook
- ACI Examples and Copybooks
- Related Literature

Call Format

Calls to EntireX Broker use the following arguments:

1. The ACI control block is the first argument.
2. The send buffer is the second argument.
3. The receive buffer is the third argument.
4. The error text buffer is the last argument. It can provide a short text of the error code, if desired. Sufficient buffer length must be supplied to allow the standard 40-byte long message to be returned by EntireX Broker. For ACI version 9 and above, the error text buffer can be greater than 40 bytes as specified in the ACI field `ERRTEXT-LENGTH`.

The send buffer and the receive buffer must always be provided. If they are not required by the selected EntireX Broker function, you can define them as CHAR(1) fields.

The API is called with a statement such as the following:

- Under all platforms with all Broker stubs:

  ```
  CALL 'BROKER' USING
     (ETCB,SEND_BUFFER,RECV_BUFFER,ERROR_TEXT);
  ```

- additionally, under z/OS, you can invoke Broker stub *CICSETB*, using the following `EXEC CICS LINK` command. The length of the COMMAREA is always 24. For example:

  ```
  EXEC CICS LINK PROGRAM('CICSETB') COMMAREA(MYCOMM) LENGTH(24)
  ```

  The COMMAREA must specify an area in working storage with the following information:

- 8-byte character field "ETBCOMM*"
- one full word containing the address of the EntireX Broker control block
- one full word containing the address of send buffer
- one full word containing the address of receive buffer
- one full word containing the address of error text buffer

If 00000000 (zeros) are returned in the `ERROR-CODE` field in the EntireX Broker control block, the operation has been performed successfully. However, function results other than 00000000 (zeros) do not necessarily indicate an error. See *Error Handling* for client and server | publish and subscribe.

Broker ACI Control Block Layout

The following table shows the Broker fields in order of the physical layout of the Broker ACI control block and provides a brief description of each field. See the actual copybook for PL/I below in *Broker ACI Control Block Copybook*.

See *Broker ACI Fields* for more information.

Broker ACI Field	PL/I Definition	Description / Related Information		API Vers.	Notes
`API-TYPE`	`ETCB_TYPE BIT(8)`	API type.	See *`API-TYPE`* and *`API-VERSION`* for client and server | publish and subscribe.	1	
`API-VERSION`	`ETCB_VERSION FIXED BIN(7)`	API version.		1	
`FUNCTION`	`ETCB_FUNCTION FIXED BIN(7)`	See *Broker ACI Functions* to perform Broker function.		1	
`OPTION`	`ETCB_OPTION FIXED BIN(7)`	See `OPTION`.		1	
	`ETCB_RESERVED CHAR(16)`	Reserved for future use.		1	1
`SEND-LENGTH`	`ETCB_SEND_LEN FIXED BINARY(31)`	Send length.	See *Using Send and Receive Buffers* for client and server | publish and subscribe.	1	
`RECEIVE-LENGTH`	`ETCB_REC_LEN FIXED BINARY(31)`	Receive length.		1	
`RETURN-LENGTH`	`ETCB_RETURN_LEN FIXED BINARY(31)`	Return length.		1	
`ERRTEXT-LENGTH`	`ETCB_ERRTEXT_LEN FIXED BINARY(31)`	Error text length.		1	
`BROKER-ID`	`ETCB_BROKER_ID CHAR(32)`	Broker ID. See *Using the Broker ID in Applications*.		1	

`SERVER-CLASS`
`SERVER-NAME`
`SERVICE`
	`ETCB_SERVER_CL CHAR(32)`
	`ETCB_SERV_NAME CHAR(32)`
	`ETCB_SERVICE_NM CHAR(32)`
		Service. See *Control Block Fields and Verbs*.		1	3, 5

`USER-ID`	`ETCB_USER_ID CHAR(32)`	User ID. See *`USER-ID`* and *`TOKEN`* for client and server | publish and subscribe.		1	
`PASSWORD`	`ETCB_PASSWD CHAR(32)`	Password. See *Authentication*.		1	4,5

Broker ACI Field	PL/I Definition	Description / Related Information	API Vers.	Notes
TOKEN	ETCB_TOKEN CHAR(32)	Reconnection token. See *USER-ID and TOKEN* for client and server | publish and subscribe.	1	3, 5
SECURITY-TOKEN	ETCB_SEC_TOKEN CHAR(32)	Security token. See *Writing Applications using EntireX Security*.	1	4, 5
CONV-ID	ETCB_CONV_ID CHAR(16)	Conversation ID. See *Conversational and Non-conversational Mode*.	1	3, 5
WAIT	ETCB_WAIT CHAR(8)	Wait value. See *Blocked and Non-blocked Broker Calls* for client and server | publish and subscribe.	1	3, 5
ERROR-CODE	ETCB_ERROR_CODE PIC'99999999'	Error code. See *Error Handling* for client and server | publish and subscribe and *Error Messages and Codes*.	1	
ENVIRONMENT	ETCB_ENVIRONMNT CHAR(32)	Environment. See *Using Internationalization* for client and server | publish and subscribe.	1	3, 5
ADCOUNT	ETCB_ADCOUNT FIXED BINARY(31)	Attempted delivery count. See *Writing Applications: Units of Work*.	2	
USER-DATA	ETCB_USER_DATA CHAR(16)	Conversation User Data. See *Managing Conversation Contexts*.	2	3, 5
not used	ETCB_MESSAGE_ID CHAR(32)	Message ID.	2	4, 5
not used	ETCB_MESS_TYPE CHAR(16)	Message type.	2	3, 5
	ETCB_LIFE_TIME CHAR(8)	Reserved for future use.	2	1, 3, 5
NEWPASSWORD	ETCB_NEW_PASSWD CHAR(32)	New password. See *Authentication*.	2	4, 5
not used	ETCB_ADAPTER_ERR CHAR(8)	Adapter error.	2	
CLIENT-UID	ETCB_CL_USER_ID CHAR(32)	Client User ID. See *Writing Applications using EntireX Security*	2	
CONV-STAT	ETCB_CONVSTAT BIT(8)	Conversation status. See *Conversational and Non-conversational Mode*.	2	
STORE	ETCB_STORE BIT(8)	Persistence or non-persistence of a UOW. See *Writing Applications: Units of Work*.	2	
	ETCB_STATUS BIT(8)	Reserved for future use.	2	1

Broker ACI Field	PL/I Definition	Description / Related Information		API Vers.	Notes
UOWSTATUS	ETCB_UOWSTATUS BIT(8)	UOW Status.	See *Writing Applications: Units of Work*.	3	3, 5
UWTIME	ETCB_UOW_TIME CHAR(8)	UOW lifetime.		3	3, 5
UOWID	ETCB_UOW_ID CHAR(16)	UOW unique identifier.		3	3, 5
USTATUS	ETCB_USER_STATUS CHAR(32)	User status.		3	
UOW-STATUS-PERSIST	ETCB_UOW_STATUSP BIT(8)	Multiplier for persistent status lifetime.		3	2
	ETCB_RESERVED3 CHAR(3)	Reserved for future use.		3	1
LOCALE-STRING	ETCB_LOCALE_STR CHAR(40)	Locale string. To be used to override or provide codepages. See *Using Internationalization* for client and server | publish and subscribe.		4	
DATA-ARCH	ETCB_DATA_ARCH BIT(8)	Data architecture. See *ICU Conversion*.		4	2
FORCE-LOGON	ETCB_FORCE_LOGON CHAR(1)	Override Broker AUTOLOGON. See *Authentication*.		6	
ENCRYPTION-LEVEL	ETCB_ENCRYPT_LVL FIXED BINARY(7)	Encryption level. See *Encryption* under *Writing Applications using EntireX Security*.		6	2
KERNELSECURITY	ETCB_KERNEL_SEC CHAR(1)	Kernel security. See *Writing Applications using EntireX Security*.		7	
COMMITTIME	ETCB_COMMIT_TIME CHAR(17)	Commit time. See *Writing Applications: Units of Work*.		7	
COMPRESSLEVEL	ETCB_COMPRESS CHAR(1)	Compression level. See *Data Compression* for client and server | publish and subscribe.		7	
	ETCB_RESERVED3 CHAR(2)	Reserved for future use.		7	1
	ETCB_RESERVED4 CHAR(4)	Reserved for future use.		7	1
UWSTAT-LIFETIME	ETCB_UOW_STAT_LIFETIME CHAR(8)	Add value for persistent status lifetime. See UWSTAT-LIFETIME.		8	
TOPIC	ETCB_TOPIC CHAR(96)	Topic name for publish and subscribe.	See *Key ACI Field Names*.	8	
PUBLICATION-ID	ETCB_PUBLICATION_ID CHAR(16)	Publication ID for publish and subscribe.		8	
CLIENT-ID	ETCB_CLIENT_ID FIXED BIN(31)	Returns to a server application the unique instance number of a client application. It is returned on receipt of a message (RECEIVE or SEND with WAIT).		9	

Broker ACI Field	PL/I Definition	Description / Related Information	API Vers.	Notes
LOG-COMMAND	ETCB_LOG_COMMAND CHAR(1)	Log the current command. See also *Programmatically Turning on Command Logging* for client and server | publish and subscribe.	9	
CREDENTIALS-TYPE	ETCB_CRED_TYPE CHAR(1)	Indicates the credentials type to be used to authenticate a user. The default is to use user ID and password. Enter "1" to specify IAF authentication.	9	
VARLIST-OFFSET	ETCB_VAR_LIST_OFF FIXED BIN(31)	Internal Software AG field.	10	
LONG-BROKER-ID-LENGTH	ETCB_LONG_BID_LEN FIXED BIN(31)	See LONG-BROKER-ID-LENGTH.	10	

Notes:

1. Reserved for future use.
2. You must set this field to a LOW-VALUE (X'00') if you do not intend to use it.
3. The field is transmitted up to the first blank or LOW-VALUE (X'00'). It is not transmitted if the first character is a blank or a LOW-VALUE (X'00').
4. All trailing LOW-VALUES (X'00') are truncated. The field is not transmitted if the entire field is a LOW-VALUE (X'00').
5. If fields are not needed for a specific command function, suppress their transmission by initializing them to blanks or LOW-VALUE (X'00').

Broker ACI Control Block Copybook

The Developer's Kit provides a copybook with the ACI control block definition. See *ACI Examples and Copybooks* for where it is provided on your platform:

```
/*
**********************************************************************
* Product        : EntireX Broker
* Copyright      : (c) Copyright Software AG 1997 - 2012.
*                  All rights reserved.
* Version        : 9.0
* File           : PLIDEF
* File Version   : $Revision: 1.1 $
* Description    : PL/I language ACI control block definitions.
*
*/

/* %OPTION LANGLVL(SAA2) */

/* - EntireX Broker API Type Constants (ETBCB_TYPE)---------------- */
DCL API_TYPE1                  BIT(8) INIT('01'B4);
DCL API_TYPE2                  BIT(8) INIT('02'B4);
DCL API_TYPE4                  BIT(8) INIT('04'B4);
DCL API_TYPE8                  BIT(8) INIT('08'B4);

/* - EntireX Broker API Version Constants (ETBCB_VERSION) ---------- */
DCL API_VERS1                  BIT(8) INIT('01'B4);
```

```
DCL API_VERS2                  BIT(8) INIT('02'B4);
DCL API_VERS3                  BIT(8) INIT('03'B4);
DCL API_VERS4                  BIT(8) INIT('04'B4);
DCL API_VERS5                  BIT(8) INIT('05'B4);
DCL API_VERS6                  BIT(8) INIT('06'B4);
DCL API_VERS7                  BIT(8) INIT('07'B4);
DCL API_VERS8                  BIT(8) INIT('08'B4);
DCL API_VERS9                  BIT(8) INIT('09'B4);
DCL API_VERS10                 BIT(8) INIT('10'B4);
DCL API_VERS_HIGHEST           BIT(8) INIT('10'B4);

/* - EntireX Broker API API Function Constants (ETBCB_FUNCTION) ---- */
DCL FCT_SEND                   BIT(8) INIT('01'B4);
DCL FCT_RECEIVE                BIT(8) INIT('02'B4);
DCL FCT_UNDO                   BIT(8) INIT('04'B4);
DCL FCT_EOC                    BIT(8) INIT('05'B4);
DCL FCT_REGISTER               BIT(8) INIT('06'B4);
DCL FCT_DEREGISTER             BIT(8) INIT('07'B4);
DCL FCT_VERSION                BIT(8) INIT('08'B4);
DCL FCT_LOGON                  BIT(8) INIT('09'B4);
DCL FCT_LOGOFF                 BIT(8) INIT('0A'B4);
DCL FCT_SET                    BIT(8) INIT('0B'B4);
DCL FCT_GET                    BIT(8) INIT('0C'B4);
DCL FCT_SYNCPOINT              BIT(8) INIT('0D'B4);
DCL FCT_KERNELVERS             BIT(8) INIT('0E'B4);
DCL FCT_LOCTRANS               BIT(8) INIT('0F'B4);
DCL FCT_SETSSLPARMS            BIT(8) INIT('10'B4);
DCL FCT_SEND_PUBLICATION       BIT(8) INIT('11'B4);
DCL FCT_RECEIVE_PUBLICATION    BIT(8) INIT('12'B4);
DCL FCT_SUBSCRIBE              BIT(8) INIT('13'B4);
DCL FCT_UNSUBSCRIBE            BIT(8) INIT('14'B4);
DCL FCT_CONTROL_PUBLICATION    BIT(8) INIT('15'B4);
DCL FCT_REPLY_ERROR            BIT(8) INIT('16'B4);

/* - EntireX Broker API Option Constants (ETBCB_OPTION) ------------ */
DCL OPT_OFF                    BIT(8) INIT('00'B4);
DCL OPT_MSG                    BIT(8) INIT('01'B4);
DCL OPT_HOLD                   BIT(8) INIT('02'B4);
DCL OPT_IMMED                  BIT(8) INIT('03'B4);
DCL OPT_QUIESCE                BIT(8) INIT('04'B4);
DCL OPT_EOC                    BIT(8) INIT('05'B4);
DCL OPT_CANCEL                 BIT(8) INIT('06'B4);
DCL OPT_LAST                   BIT(8) INIT('07'B4);
DCL OPT_NEXT                   BIT(8) INIT('08'B4);
DCL OPT_PREVIEW                BIT(8) INIT('09'B4);
DCL OPT_COMMIT                 BIT(8) INIT('0A'B4);
DCL OPT_BACKOUT                BIT(8) INIT('0B'B4);
DCL OPT_SYNC                   BIT(8) INIT('0C'B4);
DCL OPT_ATTACH                 BIT(8) INIT('0D'B4);
DCL OPT_DELETE                 BIT(8) INIT('0E'B4);
DCL OPT_EOCCANCEL              BIT(8) INIT('0F'B4);
DCL OPT_QUERY                  BIT(8) INIT('11'B4);
DCL OPT_SETUSTATUS             BIT(8) INIT('12'B4);
DCL OPT_ANY                    BIT(8) INIT('13'B4);
DCL OPT_TERMINATE              BIT(8) INIT('14'B4);
DCL OPT_DURABLE                BIT(8) INIT('15'B4);
DCL OPT_CHECKSERVICE           BIT(8) INIT('16'B4);

/* - EntireX Broker Conversation Status Constants (ETBCB_CONVSTAT) - */
DCL CONVSTAT_NEW               BIT(8) INIT('01'B4);
DCL CONVSTAT_OLD               BIT(8) INIT('02'B4);
DCL CONVSTAT_NONE              BIT(8) INIT('03'B4);
```

```
/* - EntireX Broker Store Constants (ETBCB_STORE) ------------------ */
DCL STORE_OFF                  BIT(8) INIT('01'B4);
DCL STORE_BROKER               BIT(8) INIT('02'B4);

/* - EntireX Broker Status Constants (ETBCB_STATUS) ---------------- */
DCL STAT_OFF                   BIT(8) INIT('01'B4);
DCL STAT_STORED                BIT(8) INIT('02'B4);
DCL STAT_DELIVERY_ATTEMP       BIT(8) INIT('03'B4);
DCL STAT_DELIVERED             BIT(8) INIT('04'B4);
DCL STAT_PROCESSED             BIT(8) INIT('05'B4);
DCL STAT_DEAD                  BIT(8) INIT('06'B4);

/* - EntireX Broker UOW Status Constants (ETBCB_UOW_STATUS) -------- */
DCL RECV_NONE                  BIT(8) INIT('00'B4);
DCL RECEIVED                   BIT(8) INIT('01'B4);
DCL ACCEPTED                   BIT(8) INIT('02'B4);
DCL DELIVERED                  BIT(8) INIT('03'B4);
DCL BACKEDOUT                  BIT(8) INIT('04'B4);
DCL PROCESSED                  BIT(8) INIT('05'B4);
DCL CANCELLED                  BIT(8) INIT('06'B4);
DCL TIMEOUT                    BIT(8) INIT('07'B4);
DCL DISCARDED                  BIT(8) INIT('08'B4);
DCL RECV_FIRST                 BIT(8) INIT('09'B4);
DCL RECV_MIDDLE                BIT(8) INIT('0A'B4);
DCL RECV_LAST                  BIT(8) INIT('0B'B4);
DCL RECV_ONLY                  BIT(8) INIT('0C'B4);

/* - EntireX Broker Force Logon Constants (ETBCB_FORCE_LOGON) ------ */
DCL FORCE_LOGON_NO             CHAR(1) INIT('N');
DCL FORCE_LOGON_YES            CHAR(1) INIT('Y');

/* - EntireX Broker Encryption Level (ETBCB_ENCRYPTIONLEVEL) ------- */
DCL ENCLEVEL_NONE              BIT(8) INIT('00'B4);
DCL ENCLEVEL_TO_BROKER         BIT(8) INIT('01'B4);
DCL ENCLEVEL_TO_TARGET         BIT(8) INIT('02'B4);

/* - EntireX Broker Kernel Security Constants (ETBCB_KERNEL_SEC) --- */
DCL KERNEL_SECURITY_NO         CHAR(1) INIT('N');
DCL KERNEL_SECURITY_YES        CHAR(1) INIT('Y');
DCL KERNEL_SECURITY_USER       CHAR(1) INIT('U');

/* - EntireX Broker Compression Level (ETBCB_COMPRESSLEVEL) -------- */
DCL COMPRESS_LEVEL_0           CHAR(1) INIT('0');
DCL COMPRESS_LEVEL_1           CHAR(1) INIT('1');
DCL COMPRESS_LEVEL_2           CHAR(1) INIT('2');
DCL COMPRESS_LEVEL_3           CHAR(1) INIT('3');
DCL COMPRESS_LEVEL_4           CHAR(1) INIT('4');
DCL COMPRESS_LEVEL_5           CHAR(1) INIT('5');
DCL COMPRESS_LEVEL_6           CHAR(1) INIT('6');
DCL COMPRESS_LEVEL_7           CHAR(1) INIT('7');
DCL COMPRESS_LEVEL_8           CHAR(1) INIT('8');
DCL COMPRESS_LEVEL_9           CHAR(1) INIT('9');
DCL COMPRESS_LEVEL_NO          CHAR(1) INIT('N');
DCL COMPRESS_LEVEL_YES         CHAR(1) INIT('Y');

/*-------------------------------------------------------------------*/
/* ETBCB: EntireX Broker API Control Block Definition                */
/*-------------------------------------------------------------------*/
DCL 1 ETBCB ALIGNED,
      3 ETCB_TYPE              BIT(8),
      3 ETCB_VERSION           BIT(8),
```

```
3 ETCB_FUNCTION          BIT(8),
3 ETCB_OPTION            BIT(8),
3 ETCB_RESERVED          CHAR(16),
3 ETCB_SEND_LEN          FIXED BIN(31),
3 ETCB_RECEIVE_LEN       FIXED BIN(31),
3 ETCB_RETURN_LEN        FIXED BIN(31),
3 ETCB_ERRTEXT_LEN       FIXED BIN(31),
3 ETCB_BROKER_ID         CHAR(32),     /* BROKER ID = ETB018  */
3 ETCB_SERVER_CLASS      CHAR(32),     /* SERVER CLASS        */
3 ETCB_SERVER_NAME       CHAR(32),     /* SERVER NAME         */
3 ETCB_SERVICE           CHAR(32),     /* SERVICE             */
3 ETCB_USER_ID           CHAR(32),     /* USER ID             */
3 ETCB_PASSWD            CHAR(32),     /* PASSWORD            */
3 ETCB_TOKEN             CHAR(32),     /* RECONNECTION TOKEN  */
3 ETCB_SEC_TOKEN         CHAR(32),     /* SECURITY TOKEN      */
3 ETCB_CONV_ID           CHAR(16),     /* CONV/NON CONV       */
3 ETCB_WAIT_TIMEOUT      CHAR(8),      /* BLOCKED/NON BLOCKED */
3 ETCB_ERROR_CODE        CHAR(8),      /* ERROR CLASS/NUMBER  */
3 ETCB_ENVIRONMNT        CHAR(32),     /*                     */
/* V2 ADDITIONS */
3 ETCB_ADCOUNT           FIXED BIN(31),/* ATTEMPT DELIVERY    */
3 ETCB_USER_DATA         CHAR(16),     /* USER DATA FIELD     */
3 ETCB_MESSAGE_ID        CHAR(32),     /* NOT USED BY BROKER  */
3 ETCB_MESSAGE_TYPE      CHAR(16),     /* NOT USED BY BROKER  */
3 ETCB_LIFE_TIME         CHAR(8),      /* NOT USED BY BROKER  */
3 ETCB_NEW_PASSWD        CHAR(32),     /* NEW PASSWORD        */
3 ETCB_ADAPTER_ERROR     CHAR(8),      /* ADAPTER ERROR       */
3 ETCB_CLIENT_USER_ID    CHAR(32),     /* USER-ID FROM CLIENT */
3 ETCB_CONVSTAT          BIT(8),       /* CONVERSATION STATUS */
3 ETCB_STORE             BIT(8),       /* STORE               */
3 ETCB_STATUS            BIT(8),       /* NOT USED BY BROKER  */
3 ETCB_UOW_STATUS        BIT(8),       /* STATUS OF UOW       */
/* V3 ADDITIONS */
3 ETCB_UOW_TIME          CHAR(8),      /* UOW LIFE TIME       */
3 ETCB_UOW_ID            CHAR(16),     /* UNIT OF WORK IDENT  */
3 ETCB_USER_STATUS       CHAR(32),     /* USER STATUS IN UOW  */
3 ETCB_UOW_STATUSP       BIT(8),       /* UOW STATUS PERSIST  */
3 ETCB_RESERVED2         CHAR(3),      /* ALIGNMENT           */
/* V4 ADDITIONS */
3 ETCB_LOCALE_STR        CHAR(40),     /* LOCALES             */
3 ETCB_DATA_ARCH         BIT(8),       /* FOR FUTURE USE      */
/* V6 ADDITIONS */
3 ETCB_FORCE_LOGON       CHAR(1),      /* FORCE LOGON         */
3 ETCB_ENCRYPTIONLEVEL   BIT(8),       /* ENCRYPTION LEVEL    */
/* V7 ADDITIONS */
3 ETCB_KERNEL_SEC        CHAR(1),      /* KERNEL SECURITY     */
3 ETCB_COMMIT_TIME       CHAR(17),     /* COMMIT TIME OF UOW  */
3 ETCB_COMPRESSLEVEL     CHAR(1),      /* COMPRESSION LEVEL   */
3 ETCB_RESERVED3         CHAR(2),      /* ALIGNMENT           */
3 ETCB_RESERVED4         CHAR(4),      /* ALIGNMENT           */
/* V8 ADDITIONS */
3 ETCB_UOW_STAT_LIFETIME CHAR(8),      /* UOWSTATUS LIFE: ADD */
3 ETCB_TOPIC             CHAR(96),     /* TOPIC NAME          */
3 ETCB_PUBLICATION_ID    CHAR(16),     /* PUBLICATION ID      */
/* V9 ADDITIONS */
3 ETCB_PARTNER_BID       CHAR(32),     /* PARTNER BROKER ID   */
3 ETCB_RESVD_V73_1       FIXED BIN(31),/* RESERVED FOR FUTURE */
3 ETCB_RESVD_V73_2       FIXED BIN(31),/* RESERVED FOR FUTURE */
3 ETCB_RESVD_V73_3       FIXED BIN(31),/* RESERVED FOR FUTURE */
3 ETCB_CLIENT_ID         FIXED BIN(31),/* CLIENT IDENTIFIER   */
3 ETCB_RESVD_V73_4       CHAR(32),     /* RESERVED FOR FUTURE */
3 ETCB_LOG_COMMAND       CHAR(1),      /* LOG THIS COMMAND    */
```

```
3 ETCB_CRED_TYPE         CHAR(1),      /* CREDENTIALS TYPE    */
3 ETCB_RESVD_V73_5       CHAR(32),     /* RESERVED FOR FUTURE */
3 ETCB_RESERVED5         CHAR(2),      /* ALIGNMENT           */
/* V10 ADDITIONS */
3 ETCB_VAR_LIST_OFF      FIXED BIN(31),/* VARIABLE LIST OFFSET*/
3 ETCB_LONG_BID_LEN      FIXED BIN(31);/* LEN LONG BROKER-ID  */
```

ACI Examples and Copybooks

When you begin to write your first Broker ACI program, you can use the example PLIIVP as a model for your own implementation.

Under z/OS, the Broker ACI control block copybook is contained in member PL1DEF in the mainframe source library EXX970.SRCE.

Under z/VM, the Broker ACI control block is contained in member PL1DEF of the z/VM MACLIB EXX970.MACLIB.

Related Literature

- *Writing Applications: Publish and Subscribe* - How to implement and program publish-and-subscribe applications, employing durable subscription techniques, with EntireX Broker.
- *Writing Applications: Client and Server* - How to implement and program client-and-server applications with EntireX Broker.
- *Writing Applications: Units of Work* - Describes the concept of units-of-work programming for EntireX Broker.
- *Writing Applications: Attach Server* - Describes the programming of Attach Server for EntireX Broker. It assumes you are familiar with the basics of EntireX Broker ACI programming.
- *Writing Applications: Command and Information Services* - EntireX Broker provides an API for Command and Information Services (CIS) that include the following: shutting down servers; switching trace on and off; retrieving information on clients; registered servers and services.
- *Broker CIS Data Structures* - Describes the data structures of the Command and Information Services.
- *Writing Applications using EntireX Security* - Programming aids relevant to EntireX Security programming.
- *Broker ACI Fields* - Describes the fields in the EntireX Advanced Communication Interface (ACI) that define Broker functions to be performed.
- *Broker ACI Functions* - Describes the EntireX Broker ACI functions.
- *Broker UOW Status Transition* - Contains the UOW Status transition tables for EntireX Broker.
- *Using the Broker ID in Applications* - Describes the URL-style broker ID and transport-method-style broker ID.