webMethods EntireX

Installation under IBM i

Version 9.7

October 2014

WEBMETHODS

This document applies to webMethods EntireX Version 9.7.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 1997-2014 Software AG, Darmstadt, Germany and/or Software AG USA, Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA, Inc. and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at http://softwareag.com/licenses.

Use of this software is subject to adherence to Software AG's licensing conditions and terms. These terms are part of the product documentation, located at http://softwareag.com/licenses/ and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at http://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software AG.

Document ID: EXX-INSTALL_OS400-97-20160805

Table of Contents

Installing EntireX under IBM i

Installing the EntireX RPC Server under IBM i	How to install the EntireX RPC server under IBM i.
Installing EntireX Broker Stubs under IBM i	How to install the Broker stubs under IBM i.

Related Literature

- *Administration of Broker Stubs under IBM i* in the IBM i administration documentation
- *EntireX RPC Server Return Codes* under *Error Messages and Codes*
- *Administering the EntireX RPC Server* in the IBM i administration documentation
- *Using the C Wrapper*
- *Using the COBOL Wrapper*
- *Using EntireX RPC for RPG under IBM i*
- *Using EntireX RPC for CL under IBM i*

1 Installing the EntireX RPC Server under IBM i

The EntireX RPC Server under IBM i enables you to call programs as servers, using ILE (Integrated Language Environment).

Note: The EntireX RPC server installation also includes sample programs that demonstrate how to build and use an RPC client environment on IBM i. For details, see ***Installation Verification*** and *Using the COBOL Wrapper*.

This chapter describes how to install the RPC server under IBM i.

See *Platform Coverage* in the EntireX Release Notes for full platform information.

Prerequisites for installing the RPC server are described centrally. See *IBM i Prerequisites* in the EntireX Release Notes.

The implementation under IBM i is based on the UNIX code, therefore, use the UNIX parameters for the IBM i environment, see *Setting up Broker Instances* in the UNIX administration documentation. However, some features provided under UNIX are not supported under IBM i, see ***Features Currently not Supported***.

Features Currently not Supported

Secure Sockets Layer (SSL) is currently not supported.

Installation Steps

Note: `vrsp` stands for the current version, release, service pack, and optionally a patch level.

Installation comprises the following steps:

- Step 1: Check the Product Library
- Step 2: Copy the Installation Kit to Disk
- Step 3: Verify the Contents of the *SAVF File

Step 1: Check the Product Library

A successful Broker ACI installation on your IBM i machine is a prerequisite to install the RPC server. Software AG recommends you to keep the RPC server environment in the library EXX, in which your Broker ACI was installed.

If this library does not yet exist, create it with the command `CRTLIB EXX`.

Step 2: Copy the Installation Kit to Disk

The product is delivered in a data set with the name *../OS400/EXP`vrsp`* on your EntireX installation DVD.

To copy the installation file to your IBM i disk

1 Use the command `CRTSAVF` to create an empty IBM i `*SAVF` file, named EXP *`vrsp`*, on your IBM i machine in a library of your choice.

2 Use FTP to transfer the unzipped PC file EXP *`vrsp`* to the corresponding IBM i save file using the FTP option “binary”.

Step 3: Verify the Contents of the *SAVF File

To verify the contents of the `*SAVF` file EXP*`vrsp`*

- Use the IBM i command `DSPSAVF`. The command should display the following objects:

Object	Type	Description
`XSERVER`	`*PGM`	RPC server program.
`STR_RPCSRV`	`*PGM`	Sample RPC server starting procedure.
`EXPCRTLOG`	`*PGM`	Sample procedure to create the physical log file.
`EXPRUNTIME`	`*SRVPGM`	RPC server runtime module.
`EXPMSG`	`*MSG`	EntireX Messages.
`H_EXP`	`*FILE (PF-SRC)`	Header include files for sample C programs. Rename to `H` before use.
`QCLSRC`	`*FILE (PF-SRC)`	Sources of build procedures . **Member** / **Description** `SERVER_CFG` — RPC server configuration to be used by the server startup procedure. `STR_RPCSRV` — Sample procedure to submit/start an RPC server. `EXPCRTLOG` — Procedure to create the physical log file.
`EXAMPLE`	`*SAVF`	Save file containing RPC server and RPC client examples.

Step 4: Restore the *SAVF File

To restore the *SAVF file

- Use the command `RSTLIB`:

```
RSTLIB SAVLIB(EXPvrsp) DEV(*SAVF) SAVF(yoursavlib/EXPvrsp) RSTLIB(EXX)
```

where *yoursavlib* denotes the library into which you transferred the save file during ***Step 2***.

In addition to the Broker ACI installation, the product library EXX should now contain the objects listed in ***Step 3***.

Software AG recommends you not having user objects in this library. Copy modified samples to the user libraries. Source samples mentioned here may change, therefore check all associated program objects for the latest version.

The EntireX RPC server for IBM i is now installed and ready to use.

Installation Verification

The verification is based on the EntireX RPC calculation example CALC in the IDL file *example.idl*, which is common for all platforms. The IBM i library EXAMPLE provides programs that allow you to test a COBOL RPC client and to run the RPC server with a COBOL or a C application service.

The verification comprises the following steps:

- Step 1: Restore the EXAMPLE Library
- Step 2: Verify the RPC COBOL Client
- Step 3: Verify the RPC Server using COBOL
- Step 4: Verify the RPC Server using C

Note: RPG source examples are currently not provided for the installation verification.

Prerequisite for all verifications:

The service program EXA (type *SRVPGM, the Broker ACI/stub) must be available in your library list. You can accomplish this task by adding EXX, where the Broker ACI and the RPC server are usually installed, to your library list.

Step 1: Restore the EXAMPLE Library

Before you can begin the verification, you must restore the IBM i library EXAMPLE from the save file EXAMPLE that you downloaded to library EXX during *Step 3*.

To restore the *SAVF example file, use the command RSTLIB:

```
RSTLIB SAVLIB(EXAMPLE) DEV(*SAVF) SAVF(EXX/EXAMPLE) RSTLIB(EXAMPLE)
```

The newly created library EXAMPLE should contain the following objects:

Object	Type	Used by example	Description
CALC	*PGM	COBOL server, see *Step 3: Verify the RPC Server using COBOL*	COBOL server calculation engine (created with BIND_CALC).
CALC_RPG	*PGM	RPC server	RPG Server calculation engine (created with BIND_RCALC).
CALCCLIENT	*PGM	COBOL client, see *Step 2: Verify the RPC COBOL Client*	COBOL client calculation dialog (created with BINDCCALC). Contains the modules CCALCMAIN, CCALCMENU, CCALC and RPCSRVI.
CRT_CBLMOD	*PGM	all COBOL	Procedure to compile/create COBOL modules.
CRT_C_SRV	*PGM	all C	Procedure to create modules and service programs of DEXAMPLE and EXAMPLE for the C Server.
BIND_CALC	*PGM	COBOL server, see *Step 3: Verify the RPC Server using COBOL*	Procedure to compile and bind the COBOL server.
BIND_CCALC	*PGM	COBOL client, see *Step 2: Verify the RPC COBOL Client*	Procedure to compile and bind the COBOL client.
BIND_RCALC	*PGM	RPG Server	Procedure to compile and bind the RPG Server.
STR_RPCSRV	*PGM	all Server	Procedure to start the EntireX RPC server.
X_DEXAMPLE	*SRVPGM	C Server, see COBOL client, see *Step 4: Verify the RPC Server using C*	C Server generated stub service. To use it, rename to DEXAMPLE.
X_EXAMPLE	*SRVPGM	C Server, see COBOL client, see *Step 4: Verify the RPC Server using C*	C Server implementation file. This service contains the application logic of the programs CALC, HELLO and POWER. To use it, rename to EXAMPLE.
CCALCMENU	*FILE (DSPF)	COBOL client, see *Step 2: Verify the RPC COBOL Client*	COBOL client dialog screen.

<table>
<tr><th>Object</th><th>Type</th><th>Used by example</th><th>Description</th></tr>
<tr><td>CALC</td><td>*MODULE</td><td>COBOL client, see Step 2: Verify the RPC COBOL Client</td><td>COBOL server calculation engine.</td></tr>
<tr><td>QCBLLESRC</td><td>*FILE
(PF-SRC)</td><td>all COBOL</td><td>COBOL sources for Client and Server.<table><tr><th>Member</th><th>Description</th></tr><tr><td>CALC</td><td>COBOL server calculation engine.</td></tr><tr><td>CCALCMENU</td><td>COBOL client display file.</td></tr><tr><td>CCALCMAIN</td><td>COBOL client dialog program.</td></tr><tr><td>CCALC</td><td>COBOL client stub derived from CobolClient1.tpl.</td></tr><tr><td>RPCSRVI</td><td>COBOL client Broker Service derived from CobolClient2.tpl.</td></tr><tr><td>ERXCOMM</td><td>RPC Communication Area copybook to be copied by CALCMAIN, CCALC and RPCSRVI.</td></tr></table></td></tr>
<tr><td></td><td>*FILE
(PF-SRC)</td><td>all</td><td>CL procedures for compiling, binding and building.<table><tr><th>Member</th><th>Description</th></tr><tr><td>BIND_CALC</td><td>Procedure to compile and bind the COBOL server.</td></tr><tr><td>BIND_CCALC</td><td>Procedure to compile and bind all COBOL client modules.</td></tr><tr><td>BIND_RCALC</td><td>Procedure to compile and bind the RPG server.</td></tr><tr><td>CRT_C_SRV</td><td>Procedure to create modules and service programs of DEXAMPLE and EXAMPLE for the C server.</td></tr><tr><td>CRT_CBLMOD</td><td>Compile COBOL modules.</td></tr></table></td></tr>
<tr><td>QCSRC</td><td>*FILE
(PF-SRC)</td><td>all C</td><td>Sources for C Server.<table><tr><th>Member</th><th>Description</th></tr><tr><td>DEXAMPLE</td><td>C server generated stub.</td></tr><tr><td>EXAMPLE</td><td>C server implementation file.</td></tr></table></td></tr>
<tr><td>H</td><td>*FILE
(PF-SRC)</td><td>all C</td><td>C header files. It contains the generated stub header CEXAMPLE.h.</td></tr>
<tr><td>QRPGLESRC</td><td>*FILE
(PF-SRC)</td><td>all RPG</td><td>RPG sources for server.<table><tr><th>Member</th><th>Description</th></tr><tr><td>CALC_RPG</td><td>RPG server calculation engine.</td></tr></table></td></tr>
</table>

Step 2: Verify the RPC COBOL Client

In your EntireX network environment, the RPC server program CALC must be available for calculating figures. Examples of a CALC server program are provided in C and in COBOL. You can also access the sample CALC programs installed on your IBM i computer as described in *Step 3* and *Step 4* below.

To verify the COBOL client

1 Add the sample library EXAMPLE to your library list.

2 Call program CALCCLIENT.

A menu similar to the following will be displayed:

```
Calculator Menu
-------------------------------------------------------

Operation: ± (type + - * / to calculate or
                          type .       to terminate)
Operand 1:    _____

Operand 2:    _____

Result:  ___________


Broker-ID: localhost:1971    Server: SRV1
```

3 Specify the ID of the remote Broker and the name of the server that provides the CALC program. Specify the figures you want to compute and press **Enter**. If the Broker connection fails, you will receive an appropriate error message.

The following modules/files are bound to program CALCCLIENT:

Object	Description
CCALCMAIN	The main program logic.
CCALCMENU	The menu display file.
CCALC	The client stub derived from *CobolClient1.tpl*.
RPCSRVI	The client runtime derived from *CobolClient2.tpl*.
ERXCOMM	RPC communication area copybook.
EXA	The Broker ACI (stub).

All sources are located in the file EXAMPLE/QCBLLESRC.

Use the procedure BIND_CCALC to recompile and rebind the modules.

Note: Program CALCCLIENT expects the server program CALC to be located in a library named EXAMPLE (as specified in the client stub CCALC). If your CALC program is located in a different library, you must adjust member EXAMPLE/QCBLLESRC(CCALC). Modify all occurrences of the string "EXAMPLE" to your library name and adapt the associated string length. Then compile and rebind the CALCCLIENT program.

Step 3: Verify the RPC Server using COBOL

To verify the RPC server under IBM i, you can use the COBOL server program CALC located in library EXAMPLE. When requested by a client process, it provides the four basic arithmetic operations addition, subtraction, multiplication and devision.

To verify the Server sample written in COBOL

1 Edit the RPC server configuration file EXAMPLE/QCLSRC(RPCSRV_CFG).

You must at least modify the BrokerID=*Localhost*:1971 of the remote Broker where you want to register your server and the ServerName=*SRV1* that identifies your service.

2 Start the procedure EXAMPLE/STR_RPCSRV. It will submit the RPC server named XSERVER to a batch queue.

If you want to extend the SBMJOB parameters, you can modify/recompile the procedure EXAMPLE/QCLSRC(STR_RPCSRV).

Note: The RPC server can only be started as a batch job for multithreading reasons. ALWMLTTHD=*YES is a very important parameter for allowing multiple threads. The configuration file RPCSRV_CFG described in the previous step will be passed to the XSERVER.

3 Using the IBM i command WRKACTJOB, you should find a job named RPCSERVER in your active-job list.

4 Use a calculator client process to send a request to your server. As remote client you can run a Java test generated from the Workbench *example.idl*. Or run the IBM i sample CALCCLIENT from your IBM i machine as described in ***Step 2: Verify the RPC COBOL Client***.

The RPC server will find and access the COBOL based sample program CALC in your library EXAMPLE and pass the computed result back to the client. Server stubs are not required for application servers written in COBOL and RPG.

For more details on the server access logic, see *Administering the EntireX RPC Server* in the IBM i administration documentation.

Step 4: Verify the RPC Server using C

To verify the server sample written in C

1 Edit the RPC server configuration file `EXAMPLE/QCLSRC(RPCSRV_CFG)`.

You must at least modify the `BrokerID=`*`Localhost`*`:1971` of the remote Broker where you want to register your server and the `ServerName=`*`SRV1`* that identifies your service.

2 Start the procedure `EXAMPLE/STR_RPCSRV`. It will submit the RPC server named `XSERVER` to a batch queue.

If you want to extend the SBMJOB parameters, you can modify/recompile the procedure `EXAMPLE/QCLSRC(STR_RPCSRV)`.

Note: The RPC server can only be started as a batch job for multithreading reasons. `ALWMLTTHD=*YES` is a very important parameter for allowing multiple threads. The configuration file RPCSRV_CFG described in the previous step will be passed to the `XSERVER`.

3 Using the IBM i command `WRKACTJOB`, you should find a job named "`RPCSERVER`" in your active-job list.

4 Rename the C stub `X_DEXAMPLE` to `DEXAMPLE` and the C application `X_EXAMPLE` to `EXAMPLE`.

Note: under IBM i, a service program of type `*SRVPGM` is equivalent to the UNIX term “shared library”.

For more details on the naming convention of servers and stubs written in C, see *Using the C Wrapper*.

5 Use a calculator client process to send a request to your server. As remote client you can run a Java test program generated from the Workbench example.idl.

The RPC server will search for the shared libraries D*`library`* (the server stub) and *`library`* which contains the program functions, e.g. CALC. In our sample case, *`library`* denotes EXAMPLE.

The section *Administration of the EntireX RPC Server under IBM i* describes in detail how the RPC server distinguishes between a shared library and a stubless COBOL or RPG program.

2 Installing EntireX Broker Stubs under IBM i

This chapter describes how to install the Broker stub under IBM i.

See *Platform Coverage* in the EntireX Release Notes for full platform information.

The prerequisites for installing the stub are described centrally. See *IBM i Prerequisites* in the EntireX Release Notes.

The implementation on IBM i is based on the UNIX code, therefore, use the UNIX parameters for the IBM i environment, see *Setting up Broker Instances* in the UNIX administration documentation. However, some features provided under UNIX are not supported under IBM i, see ***Features Currently not Supported*** below.

Features Currently not Supported

Secure Sockets Layer (SSL) is currently not supported.

Installation Steps

Note: *vrsp* stands for the current version, release, service pack, and optionally a patch level.

The installation comprises the following steps:

- Step 1: Create a Product Library
- Step 2: Copy the Installation Data Set to Disk
- Step 3: Verify the Contents of the *SAVF File
- Step 4: Restore the *SAVF File

Step 1: Create a Product Library

We recommend that you keep the EntireX Broker environment in an IBM i library named *EXX*.

If this library does not yet exist, create it with the IBM i command `CRTLIB EXX`.

Step 2: Copy the Installation Data Set to Disk

The product is delivered in a data set with the name *../OS400/EXAvrsp* and is stored on your EntireX installation DVD. The data set must be transferred to a *SAVF file on your machine. To do this, use the command CRTSAVF to create the empty save file EXA*vrsp* in a library of your choice. Then use FTP to transfer the unzipped PC data set to the save file using the FTP option "binary".

Step 3: Verify the Contents of the *SAVF File

To verify the content of the *SAVF file EXA*vrsp*, use the command DSPSAVF. This command will display the following objects:

Object	Type	Description
EXA	*SRVPGM	EntireX Broker ACI (stub).
EXXZCOMP	*SRVPGM	EntireX Compression Version 7.1.1.
TRANPOT	*SRVPGM	EntireX TCP/IP Transport Version 7.1.1.
X_SECUEXIT	*SRVPGM	EntireX Security Version 7.1.1 (to use, rename it). See *Using the Security Exit* in the IBM i administration documentation.
ZLIB	*SRVPGM	zlib compression.
BCOC, BCOCCBL, BCOCRPG	*PGM	Client example programs in C, COBOL and RPG.
BCOS, BCOSCBL, BCOSRPG	*PGM	Server example programs in C, COBOL and RPG.
EXABCOC	*PGM	Start BCOC client sample program (edit first; read the note under ***Test Procedures***).
EXABCOCSEC	*PGM	Start BCOC client sample program with Security (edit first; read the note under ***Test Procedures***).
EXABCOS	*PGM	Start BCOS server sample program (edit first; read the note under ***Test Procedures***).
EXABCOSSEC	*PGM	Start BCOS server sample program with Security (edit first; read the note under ***Test Procedures***).
CRT_CBLMOD	*PGM	Compile a COBOL source to an ILE module.
CRT_CMOD	*PGM	Compile a C source to an ILE module.
CRT_RPGMOD	*PGM	Compile an RPG source to an ILE module.
EXABNDPGM	*PGM	Sample: Bind user main program with the EXA service program (the Broker ACI).
EXASETENV	*PGM	Sample: Set environment variables for EXA.
EXACRTLOG	*PGM	Create a physical LOG file.
BCOC, BCOCCBL, BCOCRPG	*MODULE	Client example modules in C, COBOL and RPG.
BCOS, BCOSCBL, BCOSRPG	*MODULE	Server example modules in C, COBOL and RPG.
EXASRC	*FILE (PF-SRC)	Example source file (sample programs and procedures).
H_EXA	*FILE (PF-SRC)	Include files for sample C programs. Rename to H before use.

Object	Type	Description
QCBLLESRC	*FILE (PF-SRC)	COBOL copybooks for sample COBOL programs.
QRPGLESRC	*FILE (PF-SRC)	RPG copybooks for sample RPG programs.
QSRVSRC	*FILE (PF-SRC)	Export definitions for data and procedures.

Step 4: Restore the *SAVF File

To restore the *SAVF file use the command RSTLIB:

```
RSTLIB SAVLIB(EXAvrsp) DEV(*SAVF) SAVF(EXAvrsp) RSTLIB(EXX)
```

After you have restored the save file, the product library EXX should at least contain the objects listed in step 3.

Software AG recommends you not putting user objects in this library. Copy modified samples to the user libraries. CL samples mentioned here may change, therefore check all associated program objects for the latest version.

The EntireX Broker ACI for IBM i is now installed and ready to be bound to a user-developed application environment.

Verifying the Installation of the Broker Stubs

This package contains test programs (BCOS, BCOC) and example programs for binding (LINKPGM).

Test Programs

There are two types of programs to test the connection to the remote Broker.

- BCOS, BCOSCBL and BCOSRPG

  These programs act as servers. When they have been registered with the remote Broker, they wait for incoming calls from the client programs BCOC, BCOCCBL and BCOCRPG, respectively, and respond immediately.

- BCOC, BCOCCBL and BCOCRPG

  These programs act as clients. They attach in non-conversational mode to one service and send data to the registered server programs BCOS, BCOSCBL and BCOSRPG, respectively.

The file EXASRC contains the sources of the programs listed above; they are available in C, COBOL (...CBL) and RPG (...RPG).

Prerequisite for the test is a running remote Broker. To proceed, you need to know the Broker ID (`brokerid`), the port number (`portno`) and, optionally, the API (ACI) version of your Broker kernel (`APIversion`).

1. In two interactive sessions, set the `*CURLIB` to library EXX.
2. Check that the log file in library EXX exists. If it does not exist, create it using the example program `EXACRTLOG`.
3. In session 1, execute the command:

```
Call BCOS ('-b brokerid: portno ' '- APIversion ')
```

4. In session 2, execute the command:

```
Call BCOC ('-b brokerid: portno ' '- APIversion ')
```

The steps above use the C samples, the samples for COBOL and RPG (see the program names above) can be used with the same parameters. The COBOL and RPG samples have less screen output than the C sample.

With Broker Kernel API Version 6 and the Broker ID Localhost, the screen output of `BCOC` will look similar to this:

```
CLIENT: ETB Version 2
CLIENT: Number of repeats (10)
CLIENT: Broker ID Localhost:3930
CLIENT: Broker ID pcbal2:6099
CLIENT: Using default class ACLASS
CLIENT: Using default server ASERVER
CLIENT: Using default service ASERVICE
CLIENT: Maximum stub ACI version: 7
CLIENT: Maximum kernel ACI version: 6
CLIENT: Kernel Version 6.2.1.00 Platform UNIX

CLIENT: Using ACI version: 6
Request issued (9, 2037)
Request issued (8, 2037)
Request issued (7, 2037)
Request issued (6, 2037)
Request issued (5, 2037)
Request issued (4, 2037)
Request issued (3, 2037)
Request issued (2, 2037)
Request issued (1, 2037)
Request issued (0, 2037)
CLIENT: Test successfully performed
Press ENTER to end terminal session.
```

The screen output of BCOS will look similar to this:

```
SERVER: ACI Version 6
SERVER: Number of repeats (10)
SERVER: Broker ID Localhost:3930
SERVER: Using default class ACLASS
SERVER: Using default server ASERVER
SERVER: Using default service ASERVICE
SERVER: Maximum stub ACI version: 7
SERVER: Maximum kernel ACI version: 6
SERVER: Kernel Version 6.2.1.00 Platform UNIX

SERVER: Using ACI version: 6
Incoming data (2037, 9)
Incoming data (2037, 8)
Incoming data (2037, 7)
Incoming data (2037, 6)
Incoming data (2037, 5)
Incoming data (2037, 4)
Incoming data (2037, 3)
Incoming data (2037, 2)
Incoming data (2037, 1)
Incoming data (2037, 0)
SERVER: Test successfully performed
Press ENTER to end terminal session.
```

If it does, the test with the remote Broker was successful.

Test Procedures

To perform the test programs, you can also modify and use the sample procedures located in the source file EXASRC:

Procedure	Description	Programs called.
EXABCOC	Client test procedure	BCOC, BCOCCBL, BCOCRPG.
EXABCOCSEC	Client test procedure with Security parameters	BCOC only.
EXABCOS	Server test procedure	BCOS, BCOSCBL, BCOSRPG.
EXABCOSSEC	Server test procedure with Security parameters	BCOS only.

Note: By default, all procedures call the "C" flavor of the client and server programs, i.e. BCOC and BCOS.

To call the relevant COBOL (BCOCCBL, BCOSCBL) or RPG (BCOCRPG, BCOSRPG) programs, you must modify the procedures listed above accordingly. After having adjusted the Broker ID, Broker Version and Security parameters, you must compile the sources and bind the created modules to executable *PGM programs. For compiling, use the procedure CRT_CMOD, CRT_CBLMOD or CRT_RPGMOD. For binding, use the procedure EXABNDPGM.

All sample programs incorporate the ACI Broker control block definitions during compilation. The relevant include file and copybooks are located in:

Include source file	Copy member	Description
H	ETBCDEF	C ACI definitions for BCOS and BCOC.
	ETBCINF	Broker API definitions for Command/Info Services using C. See *Broker Command and Information Services*.
QCBLLESRC	COBDEF	COBOL ACI definitions for BCOSCBL and BCOCCBL.
	COBINF	Broker API definitions for Command/Info Services using COBOL. See *Broker Command and Information Services*.
QRPGLESRC	RPGDEF	RPG ACI definitions for BCOSRPG and BCOCRPG.

Upgrading from Version 5.3, 6.1 and 6.2

Step 1: Make a Backup

Make a backup of the existing version in library EXX.

Step 2: Install Latest EntireX ACI Version

Perform the steps listed in ***Installation Steps*** above.

All program and sample files in the library EXX will be replaced.

Step 3: Bind the Application with latest ACI Version

All application server programs that use the Broker stub (ACI) must be rebound with the server program EXA.

To perform this step, you can use the command CRTPGM (see CL example EXABNDPGM).

Using command DSPPGMREF (Display Program Reference) you can display the service programs that are currently referenced to your application.

If you use Natural RPC, you must rebind the Natural Broker Wrapper program NXWRAPPR located in your library NXW*vrs*.