Administering the BS2000/OSD Batch RPC Server

The EntireX BS2000/OSD Batch RPC Server allows standard RPC clients to communicate with RPC servers on the operating system BS2000/OSD. It supports the programming languages COBOL and C.

This chapter covers the following topics:

- Customizing the RPC Server
- Configuring the RPC Server
- Locating and Calling the Target Server
- Starting the RPC Server
- Stopping the RPC Server
- Activating Tracing for the RPC Server

Customizing the RPC Server

The following elements are used for setting up the BS2000/OSD Batch RPC Server:

- Common Runtime Environment (CRTE)
- Configuration File
- Start Procedure

Common Runtime Environment (CRTE)

When the BS2000/OSD Batch RPC Server calls COBOL or C server programs, the BS2000/OSD Common Runtime Environment (CRTE) is loaded dynamically into the corresponding address space of the worker task.

There is no need to bind the CRTE statically to the called server object modules. If this is needed for any reason, the CRTE must be linked as a subsystem. All entries must be hidden to prevent duplicates. Linking the CRTE statically will occupy resources and slow down the load time of the server object modules.

The CRTE is not delivered with this package. For a detailed description, see the *CRTE (BS2000/OSD) User's Guide*.

Configuration File

The name of the delivered example configuration file is "RPC-CONFIG". The configuration file contains the configuration for the BS2000/OSD Batch RPC Server. The following settings are important:

- connection information such as broker ID, server address (class, name, service)
- location and usage of server-side mapping container, see *Usage of Server Mapping Files*
- scalability parameters
- trace settings
- etc.

For more information see *Configuring the RPC Server*.

Start Procedure

The name of the start S-procedure for the BS2000/OSD Batch RPC Server is "START-RPC-SERVER". The start procedure contains the following:

- the location of the Common Runtime Environment (CRTE)
- the target server library name of the called COBOL or C server
- the configuration file used; see *Configuration File*
- etc.

Configuring the RPC Server

The following rules apply:

- Underscored letters in a parameter indicate the minimum number of letters that can be used for an abbreviated command.

  For example, in `brokerid=localhost`, `brok` is the minimum number of letters that can be used as an abbreviation, i.e. the commands/parameters `broker=localhost` and `brok=localhost` are equivalents.

Parameter	Default	Values	Req/ Opt
`brokerid`	`localhost`	Broker ID used by the server. See *Using the Broker ID in Applications*. Example: `brokerid=myhost.com:1971`	R
`class`	`RPC`	Server class part of the server address used by the server. The server address must be defined as a service in the broker attribute file (see *Service-specific Attributes* under *Broker Attributes*). Case-sensitive, up to 32 characters. Corresponds to `CLASS`. Example: `class=MyRPC`	R

Parameter	Default	Values	Req/ Opt
codepage	no codepage transferred	Depending on the internationalization approach, the codepage (locale string) where incoming data is provided to the COBOL server. Conversely, the COBOL server must provide outgoing data in the given codepage, otherwise unpredictable results occur. See *What is the Best Internationalization Approach to use?* under *Internationalization with EntireX* for information on which internationalization approach requires a codepage (locale string). By default, no codepage is transferred to the broker. For the most popular internationalization approach, *ICU Conversion*, the correct codepage (locale string) must be provided. This means it must: • follow the rules described under *Locale String Mapping* • be a codepage supported by the broker • be the codepage used in your environment for file and terminal IO, otherwise unpredictable results may occur. Example: `codepage=EDF041`	R

compresslevel	N	Enforce compression when data is transferred between broker and server. See *Data Compression in EntireX Broker*.
		
		`compresslevel= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8| 9 | Y | N`
		
		0-9 0=no compression 9=max. compression
		N No compression.
		Y Compression level 6.
		
		Example:
		`compresslevel=6`
			O

deployment	NO	Activates the deployment service, see *Deployment Service*. Required to use the Server Mapping Deployment Wizard. See *Server Mapping Deployment Wizard* in the EntireX Workbench documentation. **YES** Activates the deployment service. The RPC server registers the deployment service in the broker. **NO** The deployment service is deactivated. The RPC server does not register the deployment service in the broker. Example: `deployment=yes`	O

Parameter	Default	Values	Req/ Opt
encryptionlevel	0	Enforce encryption when data is transferred between client and server. Requires EntireX Security. See ENCRYPTION-LEVEL under *Broker ACI Fields*. **0** Encryption is enforced. **1** Encryption is enforced between server and broker kernel. **2** Encryption is enforced between server and broker kernel, and also between client and broker. Example: encryptionlevel=2	O
init_exit		Initialization exit. The BS2000/OSD Batch RPC Server provides user exits that allow you to plug in code during initialization and to terminate RPC worker tasks. This parameter specifies the name of an executable module that is loaded and executed during initialization of each worker task. See also term_exit. Example: init_exit=myExit	O
extractor	NO	The extractor service is a prerequisite for remote extractions. See *Extractor Service*. extractor=YES|NO Example: extractor=yes	O
logon	YES	Execute broker functions LOGON/LOGOFF in worker threads. Must match the setting of the broker attribute AUTOLOGON. Reliable RPC requires logon set to YES. See *Reliable RPC*. **NO** No logon/logoff functions are executed. **YES** Logon/logoff functions are executed. Example: logon=no	O

Parameter	Default	Values	Req/ Opt
marshalling	COBOL	The BS2000/OSD Batch RPC Server can be configured to support either COBOL or C. See also *Locating and Calling the Target Server*. marshalling=(LANGUAGE=COBOL|C) **COBOL** The BS2000/OSD Batch RPC Server supports COBOL. The COBOL servers are called directly without a server interface object. The COBOL server modules may be compiled as OM or LLM modules. So-called server mapping files are used to call the COBOL server correctly if one is available. See *Usage of Server Mapping Files*. **C** The BS2000/OSD Batch RPC Server supports C. The modules are called using a server interface object built with the *C Wrapper*.	O
password	no default	Password for broker logon. Case-sensitive, up to 32 characters. For more information see broker ACI control block field PASSWORD. Example: password=MyPwd	O
restartcycles	15	Number of restart attempts if the broker is not available. This can be used to keep the BS2000/OSD Batch RPC Server running while the broker is down for a short time. A restart cycle will be repeated at an interval which is calculated as follows: timeout + ETB_TIMEOUT + 60 seconds where timeout is the RPC server parameter (see this table), and ETB_TIMEOUT is the environment variable (see *Environment Variables in EntireX*) When the number of cycles is reached and a connection to the broker is not possible, the RPC server stops. Example: restartcycles=30	O
servername	SRV1	Server name part of the server address used by the server. The server address must be defined as a service in the broker attribute file. See *Service-specific Attributes* under *Broker Attributes*. Case-sensitive, up to 32 characters. Corresponds to SERVER of the broker attribute file. Example: servername=mySrv	R

Parameter	Default	Values	Req/ Opt
service	CALLNAT	Service part of the server address used by the server. The server address must be defined as a service in the broker attribute file. See *Service-specific Attributes* under *Broker Attributes*. Case-sensitive, up to 32 characters. Corresponds to SERVICE attribute of the broker attribute file. Example: service=MYSERVICE	R
smhport	0	The port where the server listens for commands from the System Management Hub (SMH). If this port is 0 (default), no port is used and management by the SMH is disabled. Example: smhport=3001	O
svm	PREFERRED	Usage of server mapping files. See *Server-side Mapping Files in the RPC Server*. SVM=PREFERRED|NO **PREFERRED** This setting is to support COBOL server programs that do not have server-side mapping, plus COBOL server programs built with a server-side mapping file. If you use server-side mapping files, the server-side mapping container must be installed and configured. See *Step 1: Define a Server-side Mapping Container* in the BS2000/OSD Installation documentation. There are also client-side mapping files that do not require configuration here; see *Server Mapping Files for COBOL* in the EntireX Workbench documentation. **NO** Server-side mapping files are not used. Example for BS2000/OSD: SVM=NO See also *Usage of Server Mapping Files*.	O
term_exit		Termination exit. The BS2000/OSD Batch RPC Server provides user exits that allow you to plug in code during initialization and terminate RPC worker tasks. This parameter specifies the name of an executable module that is loaded and executed during termination of each worker task. See also init_exit. Example: term_exit=myExit	O
timeout	60	Timeout in seconds, used by the server to wait for broker requests. See broker ACI control block field WAIT for more information. Also influences restartcycles. Example: timeout=300	O

Parameter	Default	Values	Req/ Opt
tracedestination	ERXTrace.*nnn*.log	Trace output is written to SYSOUT.	O

tracelevel	None	Trace level for the server. See also *Activating Tracing for the RPC Server*.
		`tracelevel = None | Standard | Advanced | Support`
		None No trace output.
		Standard For minimal trace output.
		Advanced For detailed trace output.
		Support This trace level is for support diagnostics and should only be switched on when requested by Software AG support.
		Example:
		`tracelevel=standard`
			O

userid	ERX-SRV	Used to identify the server to the broker. See broker ACI control block field USER-ID. Case-sensitive, up to 32 characters. Example: `userid=MyUid`	R

Parameter	Default	Values	Req/ Opt
workermodel	SCALE,1,3,slowshrink	The BS2000/OSD Batch RPC Server can be configured to • adjust the number of worker threads to the current number of client requests: workermodel=(SCALE,*from*,*thru* [,slowshrink | fastshrink]) • use a fixed number of worker threads: workermodel=(FIXED,*number*) **FIXED** A fixed *number* of worker threads is used by the BS2000/OSD Batch RPC Server. **SCALE** The number of worker threads is adjusted to the current number of client requests. With the *from* value, the minimum number of active worker threads can be set. The *thru* value restricts the maximum number of worker threads. **slowshrink** The RPC server stops all worker threads not used in the time specified by the timeout parameter, except for the number of workers specified as minimum value. This is the default if SCALE is used. **fastshrink** The RPC server stops worker threads immediately as soon as it has finished its conversation, except for the number of workers specified as minimum value. Example: workermodel=(SCALE,2,5)	O

Locating and Calling the Target Server

Target server programs are loaded dynamically, using the BS2000 BLSLIB chain. The target server library name needs to be set up as `PROGRAM-LIB` in the parameter declaration section of the START-RPC-SERVER S-procedure, see *Start Procedure*. Different mechanisms are used depending on the language:

- COBOL
- C

COBOL

The approach used to derive the COBOL object module name for the RPC server depends on whether server mapping is used or not. See *Usage of Server Mapping Files* for an introduction.

1. If the RPC client sends a client-side type of server mapping with the RPC request, this server mapping is used first.
2. If no server mapping is available from step 1 above, and if server-side type of server mapping is used, the IDL library and IDL program names are used to form a key to locate the server mapping in the server-side mapping container. If a server mapping is found, this is then used.
3. If a server mapping is available from step 1 or 2 above, the COBOL object module name of the RPC server is derived from this mapping. In this case the IDL program name can be different to the COBOL object module name if it is renamed during wrapping process (see *Customize Automatically Generated Server Names*) or during the extraction process in the *COBOL Mapping Editor*.
4. If no server mapping is used at all, the IDL program name is used as the COBOL object module name of the RPC server (the IDL library name is ignored).

See also *Scenario I: Calling an Existing COBOL Server* or *Scenario II: Writing a New COBOL Server*.

To use the Batch RPC Server with COBOL

1. Make sure that all target server programs called as RPC servers
   - are COBOL object modules
   - use COBOL calling conventions
2. Configure the parameter `marshalling` for COBOL, for example:

   ```
   marshalling=COBOL
   ```

C

To use the Batch RPC Server with C

1. Make sure that all target server programs called as RPC servers

- are C object modules
- use C calling conventions

2. Configure the parameter `marshalling` for C, for example:

```
marshalling=C
```

See *Scenario III: Writing a New C Server*.

Starting the RPC Server

To start the BS2000/OSD Batch RPC Server

- Use the following SDF command:

```
/ENTER-PROCEDURE *LIB(LIB=EXP811.JOBS,ELE=START-RPC-SERVER), -
/JOB-NAME=RPCMAIN,LOG=*NO
```

Stopping the RPC Server

To stop the BS2000/OSD Batch RPC Server using System Management Hub

- Use the RPC server agent in the SMH to stop the BS2000/OSD Batch RPC Server.

To stop the BS2000/OSD Batch RPC Server from a privileged user ID

- Enter the command:

```
/INFORM-PROGRAM MSG='STOP',JOB-IDENTIFICATION=*TSN(TSN=tsn)
```

where *tsn* is the task number associated with the BS2000/OSD Batch RPC Server main task (in the example above the TSN of `RPCMAIN`)

All other tasks that were created as a result of starting the batch RPC server will be stopped automatically.

To stop the BS2000/OSD Batch RPC Server from an operator console

- Enter the command:

```
/INTR tsn,STOP
```

where *tsn* is the task number associated with the BS2000/OSD Batch RPC Server main task (in the example above the TSN of `RPCMAIN`)

All other tasks that were created as a result of starting the batch RPC server will be stopped automatically.

To stop the BS2000/OSD Batch RPC Server from a non-privileged user ID

- Use S-procedure STOP-RPC-SERVER in EXP811.JOBS.

Startup Parameter	Description	Default

BROKER-ID	Depending on the communication method, the broker ID can be specified in two different formats:
	• **TCP Transport Method**
	`ip:port:TCP`
	where *ip* is the address or DNS host name,
	port is the port number that EntireX Broker is listening on, and
	TCP is the protocol name
	• **NET Transport Method**
	`ETBnnn:SVCmmm:NET`
	where *nnn* is the ID under which EntireX Broker is connected to the Adabas ID table,
	mmm is the SVC number under which the Adabas ID table can be accessed, and
	NET is the protocol name
		none

CLASS	The class name under which the RPC server is registered at the EntireX Broker.	RPC
SERVER	The server name under which the RPC server is registered at the EntireX Broker.	SRV1
SERVICE	The service name under which the RPC server is registered at the EntireX Broker.	CALLNAT
USERID	If EntireX Broker is running with EntireX Security, a user ID needs to be supplied	none
PASSWORD	If EntireX Broker is running with EntireX Security, a password needs to be supplied	none
EXX-JOBS	EntireX Broker jobs library	EXX811.JOBS
EXX-LIB	EntireX Broker module library	EXX811.LIB
WAL-MOD	WAL module library	WAL826.MOD

Set the broker ID in the `PARAMETER-DECLARATION` section and enter following command:

```
/CALL-PROCEDURE (EXP811.JOBS, STOP-RPC-SERVER)
```

Activating Tracing for the RPC Server

To switch on tracing for the RPC server

- Set the parameter TRACELEVEL in S-element RPC-CONFIG in EXP811.JOBS.

  To evaluate the return codes, see *Error Messages and Codes*.