BEA eLink

Business Process Option

Operation and Maintenance Guide

BEA Business Process Option 1.2
Document Edition 1.2
February 2000

Document Edition	**Part Number**	**Date**	**Software Version**
1.2	870-001020-001	February 2000	BEA Business Process Option 1.2

Contents

About This Guide

This document provides detailed instructions for operating and maintaining the BEA Business Process Option system.

Who Should Read This Document

This document is intended for system administrators responsible for operating and maintaining the BEA Business Process Option system.

How This Document Is Organized

The *Operation and Maintenance Guide* is organized as follows:

- Chapter 1, “Maintaining the eLink Business Process Option”
- Chapter 2, “eLink Business Process Option Daemon Manager”
- Chapter 3, “Using the Batch Registry”
- Appendix A, “Batch Registry Command Reference.”

How to Use This Document

This document, the BEA *eLink Business Process Option Operation and Maintenance Guide*, is provided as a virtual print document in Adobe Acrobat.

Documentation Conventions

The following documentation conventions are used throughout this document.

Convention	Item
boldface text	Indicates terms defined in the glossary.
Ctrl+Tab	Indicates that you must press two or more keys simultaneously.
italics	Indicates emphasis or book titles.

`monospace text`	Indicates code samples, commands and their options, data structures and their members, data types, directories, and file names and their extensions. Monospace text also indicates text that you must enter from the keyboard.
	Examples:
	`#include <iostream.h> void main ( ) the pointer psz`
	`chmod u+w *`
	`\tux\data\ap`
	`.doc`
	`tux.doc`
	`BITMAP`
	`float`

`monospace boldface text`	Identifies significant words in code. *Example*: `void` **`commit`** `( )`
`monospace italic text`	Identifies variables in code. *Example*: `String` *`expr`*
UPPERCASE TEXT	Indicates device names, environment variables, and logical operators. *Example*s: LPT1 SIGNON OR

Convention	Item
`{ }`	Indicates a set of choices in a syntax line. The braces themselves should never be typed.
`[ ]`	Indicates optional items in a syntax line. The brackets themselves should never be typed. *Example*: `buildobjclient [-v] [-o name ] [-f file-list]... [-l file-list]...`
`|`	Separates mutually exclusive choices in a syntax line. The symbol itself should never be typed.
`...`	Indicates one of the following in a command line: ■ That an argument can be repeated several times in a command line ■ That the statement omits additional optional arguments ■ That you can enter additional parameters, values, or other information The ellipsis itself should never be typed. *Example*: `buildobjclient [-v] [-o name ] [-f file-list]... [-l file-list]...`
. . .	Indicates the omission of items from a code example or from a syntax line. The vertical ellipsis itself should never be typed.

Related Documentation

The following sections list the documentation provided with the eLink software, and other publications related to its technology.

eLink Business Process Option Documentation

The eLink Business Process Option information set consists of the following documents:

BEA eLink Business Process Option Release Notes

BEA eLink Business Process Option User's Guide

BEA eLink Business Process Option Server Installation Guide

BEA eLink Business Process Option Client Installation Guide

BEA eLink Business Process Option Operation and Maintenance Guide

Note: These documents are online at the BEA Web site. The BEA Business Process Option CD set includes all the documents in both HTML and Adobe Acrobat PDF format. You can use the Adobe Acrobat Reader to print all or a portion of each document.

BEA Publications

The following BEA publications which cover the eLink platform technology in depth are also available in the same formats as the Business Process Option documentation set:

TUXEDO System 6.5 Administration Guide

TUXEDO System 6.5 Administration Guide to the Web-Based GUI

TUXEDO System 6.5 Reference Manual

Other Publications

For more information about the eLink platform technology, refer to the following books:

3-Tier Client/Server at Work (Edwards)

The TUXEDO System (Andrade, Carges, Dwyer, Felts)

The BEA eLink Business Process Option incorporates third-party process engine technology. The relevant documentation is directly incorporated within the eLink Business Process Option documentation set. This information should be sufficient; however, the eLink Business Process Engine also contains relevant third-party documentation. Please note that neither the third-party documentation nor the usage it describes are directly supported by BEA Systems, Inc.

Contact Information

The following sections provide information about how to obtain support for the documentation and software.

Customer Support

If you have any questions about this version of the BEA eLink Business Process Option, or if you have problems installing and running the BEA eLink Business Process Option, contact BEA Customer Support through BEA WebSupport at `www.beasys.com`. You can also contact Customer Support by using the contact information provided on the Customer Support Card, which is included in the product package.

When contacting Customer Support, be prepared to provide the following information:

- Your name, e-mail address, phone number, and fax number
- Your company name and company address

- Your machine type and authorization codes
- The name and version of the product you are using
- A description of the problem and the content of pertinent error messages

CHAPTER

1 Maintaining the eLink Business Process Option

The following sections describe routine maintenance procedures for the Business Process Option:

- Overview of Administrative Procedures
- Starting, Stopping, and Configuring eLink Business Process Option
- Using Business Process Option Administration Tools
- Using Database Tools
- Setting Configuration Options
- Managing Status and Audit Logs
- Maintaining the Repository
- Optimizing the Business Process Option Database
- Moving Work Between eLink Business Process Option Servers

Overview of Administrative Procedures

Once the eLink Business Process Option is running, perform the following administrative procedures (using the `icadmin` user account):

- Add and remove new users and pools.
- Grant access privileges to users.
- Create triggers for event notification, policy enforcement, or process control.
- Export and import processes to move work between servers.

As the Administrator, you are also responsible for the following routine maintenance procedures:

- Configuring and booting the eLink Business Process Option
- Managing the Business Process Option Server
- Monitoring the status of the Repository Server
- Managing the status and audit logs
- Performing troubleshooting
- Configuring security levels for eLink Business Process Option users and actions

Starting, Stopping, and Configuring eLink Business Process Option

Most eLink Business Process Option operations can be performed using Tuxedo utilities, such as `tmadmin`, `tmloadcf`, `tmboot`, `tmshutdown`, and/or the BEA Web Administration Console. For further information, see your eLink platform documentation.

Using Business Process Option Administration Tools

In addition to the standard Tuxedo utilities, eLink Business Process Option provides administration utilities to help you perform maintenance tasks on the database and the Repository Server. These utilities are:

- `Batch Registry`
- `IcAuditManage`
- `IcStatusManage`
- `IcRepManage`
- `IcRepCheck`
- `IcRecreateAttrViews`

Using the Batch Registry Utility

The `Batch Registry` utility is an application that lets you manage the business process environment. It provides management functions that let you grant privileges to users, add users to the system, and create documents, pools, repositories, and other objects.

Note: In eLink Business Process Option, the Batch Registry accommodates whatever Tuxedo security settings you have. Depending on the security level you have set, you may be prompted for one or two user name/password combinations when you launch the Batch Registry.

Using eLink Business Process Option Maintenance Utilities

Several maintenance routines should be performed using the following utilities:

Utility	Usage
`IcAuditManage`	A utility that enables you to view audit log information unconditionally or conditionally based on a category, time period, or combination of these. Delete audit log entries unconditionally or conditionally.
`IcStatusManage`	A utility that enables you to view status log information unconditionally, or conditionally based on facility, severity, time period, or a combination of these. Delete status log entries unconditionally or conditionally.
`IcRepManage`	A utility that enables you to view and delete unused files in the eLink Business Process Option Repository.
`IcRepCheck`	A utility that enables you to check the integrity of repositories against the database, check the integrity of the database against repositories, check the integrity of a document, and check the integrity of a content file.

Using Database Tools

You can get information about the system by examining eLink Business Process Option database views with your database query language. You can use the standard views supplied, extend the standard views with additional information, or create new views.

If you add attributes to object classes, you can update the corresponding database view to include these new attributes with the `IcRecreateAttrViews` utility.

Setting Configuration Options

The eLink Business Process Option has several configuration options that are independent of Tuxedo configuration information. These options relate to the database and status audit logs.

`IcEnvFile` is a text file located in `$IC_HOME/config`.

`IcEnvFile` contains a series of lines in the following format:

```
CONFIGURATION_OPTION_NAME=VALUE
```

The following table lists the configuration options in `IcEnvFile`

Configuration option	**Description**
`IC_ACTION_POLL_INTERVAL`	Polling interval for processing triggered actions, from 1 to 300 seconds. The default is **60**.
`IC_ACTION_RPC_RETRY_INTERVAL`	Retry interval for registered trigger actions, from 900 to 86400 seconds. The default is **10800**.
`IC_ACTION_RPC_TIMEOUT`	Timeout value for registered trigger actions, from 15 to 300 seconds. The default is **60**.
`IC_AUDIT_LOG`	If an optional Audit Log File exists in addition to the Audit Log table, this variable is set to the full path name of the audit log file. The Audit Log File is in `$IC_HOME/logs` on the host machine. Default value is **NOT_USED**.
`IC_AUDIT_MODE`	Default is **IC_EVENT_AUDIT_DATABASE**. Do not change the value of this option.
`IC_EVENT_POLL_INTERVAL`	Polling interval for checking temporal events (for example, `TASK_LATE_START`, `TASK_OVERDUE`), from 60 to 86400 seconds. The default is **3600**.
`IC_HOST_MACHINE`	The host system on which the eLink Business Process Option server is running.
`IC_RDBMS`	Type of database for this installation (for example, **ORACLE**).

Configuration option	Description
`IC_SERVER_NAME`	Logical name of the eLink Business Process Option server instance (established during installation).
`IC_STATUS_LOG`	Full path name of the file containing the Process Option status output. The log file is in `$IC_HOME/logs` on the host machine.

Managing Status and Audit Logs

This section explains how to examine and interpret log entries and manage log file size on an eLink Business Process Option server.

- The Business Process Option Status Log reports errors or other significant events encountered by the eLink Business Process Option server.
- The Business Process Option Audit Log lists every event processed by the eLink Business Process Option server.

You can use the Business Process Option Status and Audit logs to:

- View audit log information unconditionally or conditionally based on a category, time period, or combination of these.
- View status log information unconditionally or conditionally based on facility, severity, time period, or a combination of these.
- Delete audit and status log entries unconditionally or conditionally.

eLink Business Process Option maintains the following log files in the `$IC_HOME/logs` directory on the eLink Business Process Option host machine:

Log File	Description
`audit_manage.log`	Audit log file
`ic_batch_registry.log`	Batch Registry log file

`IcCliMgr.err`	Error log for Daemon Manager
`CliMgr.log`	Client Manager log file
`IcEvtAct.err`	Error log for EventAction daemon
`IcGenWorker.err`	Error log for IcGen_*tag* services
`IcJobWorker.err`	Error log for IcJob_*tag* service
`IcRepCheck.log`	Log file for `IcRepCheck` utility
`IcRepManage.log`	Log file for IcRepManage utility
`status.log`	The Status log file
`ULOG.`*`mmddyy`*	Log file for eLink administrative information

Managing Status Information with IcStatusManage

eLink Business Process Option records status information to the Status log on an ongoing basis.

This status information provides:

- Verification that the installation is running smoothly
- Notification of problem areas or server startup failures
- Feedback in the event of a system failure

Status information increases in accordance with user activity. If left unchecked, this information can use up file system space. Examine the Status Log regularly with the `IcStatusManage` utility.

`IcStatusManage` has two operating modes:

Interactive mode	Lets you examine and remove log entries with a menu-based interface.
Command mode	Bypasses the menu interface and lets you examine and remove log entries with command line options.

This utility cannot be used to examine and maintain eLink Business Process Option audit information. You must use the `IcAuditManage` utility and standard UNIX file system commands to manage audit information.

Using IcStatusManage in Interactive Mode

`IcStatusManage` lets you list or remove log file entries by date, severity, or facility (application component).

During interactive processing, you specify the kind of operation you want to perform and respond to a series of menus to further qualify the operation. Every menu includes a `Quit` option that returns you to the previous menu. To exit `IcStatusManage`, select `Quit` from the main menu.

To use IcStatusManage in interactive mode

1. At the prompt, enter the following command:

   ```
   IcStatusManage
   ```

2. At the log file prompt, enter the full path name of the log file to be examined and press `RETURN`. This displays the `IcStatusManage` main menu shown below:

   ```
   Main Menu
   1. List Status Log Entries
   2. Remove Status Log Entries
   3. Quit
   ```

3. To examine log entries, choose List Status Log Entries. To remove entries from the log file, choose Remove Status Log Entries. `IcStatusManage` displays the first of a series of submenus where you specify:

   Log entry severity

   Facility from which the log entry originated

   Date of log entries

   After you enter your selections, `IcStatusManage` either displays or removes the specified log entries and returns to the Date menu.

4. Enter a different date to list or remove additional entries or select `Quit` to return to the previous menu.

Using IcStatusManage in Command Mode

The command interface for `IcStatusManage` lets you examine or remove log file entries with a series of command line options.

Syntax

```
IcStatusManage [ -s log_pathname ] [ -l | -r ]
[ -warn | -error | -fatal ]
[ -before date_time | -after date_time ]
[ -fac facility_name ]
```

Parameters

`-s` *`log_pathname`*

Full path name of the status log file to be examined. You can specify just the log file name if the file resides in $IC_HOME/logs. If you omit this parameter, IcStatusManage executes in interactive mode.

`-l | -r`

Determines the operation to be performed.
-l lists log file entries; this is the default.
-r removes log file entries.

`-warn | -error | -fatal`

In combination with the -l and -r parameters, further narrows the selection of status log entries to those marked WARNING, ERROR, or FATAL. The default is to operate on all status records.

`-before` *`date_time`* `|`
`-after` *`date_time`*

In combination with the -l and -r parameters, list or remove status log entries dated earlier than (-before) or later than (-after) date_time. The date_time value must be entered in quotes, in the format:

```
“mm/dd/yy hh:mm:ss”
```

`-fac` *`facility_name`*

In combination with the -l and -r parameters, lists or removes status log entries originating from the specified facility. The `facility_name` can be any of the facilities listed in the following table. The default is to operate on entries originating from all facilities

The following table displays the facility names.

Status Log Facility	Description
`IC_ADMIN`	eLink utility software errors
`IC_CLIENT`	User application errors
`IC_DBMS`	InConcert database errors
`IC_FILE_SYSTEM`	File system errors
`IC_GUI`	Graphical user interface errors
`IC_IC`	Process Option or general errors
`IC_NETWORK`	Network-related errors
`IC_OS`	Operating system errors
IC_TUXEDO	Tuxedo errors
`IC_WINDOW_SYSTEM`	Window system errors

Example

```
IcStatusManage -s status_log -r -warn
```

Managing Audit Information with IcAuditManage

The eLink Business Process Option captures audit (event) information to provide a tracking mechanism and history of process activity. Process activity is stored in the Audit Log Table in the Business Process Option database. An optional Audit Log file contains the same information in ASCII format.

Event information grows in direct proportion to user activity. If left unchecked, this information will continue to grow, using up file system space and possibly degrading system performance. As a result, you should examine the Audit Log regularly and clean it up using the `IcAuditManage` utility.

Note: The Audit Log Table stores iteration information about active eLink processes. Therefore, deleting table entries may cause a loss of iteration data. Use caution when deleting entries.

Like *`IcStatusManage`*, *`IcAuditManage`* has two modes:

Interactive mode	Lets you examine and remove event records with a menu-based interface.
Command mode	Bypasses the menu interface and lets you examine and remove event records with command line options.

Note: The `IcAuditManage` utility places a heavy load on the server and can degrade system performance when it is run while the system is in active use. If this is a concern, we recommend running `IcAuditManage` during off hours.

The eLink Business Process Option must be running when using `IcAuditManage`.

Note: Output written to the audit log reflects Greenwich Mean Time (GMT). All date and time input from users should be specified for the individual user's time zone.

Using IcAuditManage in Interactive Mode

`IcAuditManage` can list or remove event records by event type, user, or process. During interactive processing, you specify the kind of operation you want to perform (list or remove) and respond to a series of menus to further qualify the operation.

Every menu includes a `Quit` option that returns you to the previous menu. To exit `IcAuditManage`, select `Quit` from the main menu.

To use IcAuditManage in interactive mode

1. At the prompt, enter the following command:

   ```
   IcAuditManage -icclient user/password@server
   ```

   For example:

   ```
   icdba/icdba@manatee.
   ```

2. The `IcAuditManage` main menu is displayed:

   ```
   Main Menu
   1. List Audit Table Entries
   2. Remove Audit Table Entries
   3. Quit
   ```

3. To examine Audit Table entries, choose List Audit Table Entries. To remove entries from the Audit Table, choose Remove Audit Table Entries. `IcAuditManage` displays the first of a series of submenus where you specify one or more of the following:

   Kind of event

   User name

   Process name

   Date of entries

4. After you make your selections, `IcAuditManage` either displays or removes the specified event records and returns to the Date menu. Enter a different date to list or remove additional entries or select `Quit` to return to the previous menu.

Using IcAuditManage in Command Mode

The command interface for `IcAuditManage` lets you examine or remove Audit Log Table entries with a series of command line options.

Syntax

```
IcAuditManage -icclient client_data [ -ictimeout seconds ]
[ -r | -l ] [ -before date_time | -after date_time ]
[ -event event_type | -user user_name | -job job_name ]
```

Parameters

`-icclient` *`client_data`*

Client data of the form:

`user`*`/password@server`*

For example: `icdba/icdba@manatee`.

`-ictimeout` *`seconds`*

Timeout interval, in seconds. The default timeout is 120 seconds.

`-r | -l`

-r removes entries from the Audit Log Table. -l lists entries in the Audit Log Table; this is the default.

`-before` *`date_time`* `| -after` *`date_time`*

In combination with the -l and -r parameters, list or remove event records dated earlier than (-before) or later than (-after) date_time. The -after parameter cannot be specified if you also specify -r. The date_time value must be entered in quotes, in the format:

```
“mm/dd/yy hh:mm:ss”
```

`-event` *`event_type`* `|` `-user` *`user_name`* `|` *`-job job_name`*

In combination with the -l and -r parameters, list or remove event records by event type, user, or process.

`-event`

Narrows the operation to events of a particular kind.

`-user`

Narrows the operation to events produced by a particular Business Process Option user.

`-job`

Narrows the operation to events produced by a particular process.

In combination with the `-l` and `-r` parameters, list or remove event records by event type, user, or process.

`-event`	Narrows the operation to events of a particular kind.
`-user`	Narrows the operation to events produced by a particular user.
`-job`	Narrows the operation to events produced by a particular process.

Managing the Optional Audit Log file

The optional Audit Log file is created during Process Option startup. The Audit Log file is created if the `IC_AUDIT_LOG` parameter is set to a full path name in `IcEnvFile`.

The Audit Log file contains the text equivalent of the Audit Log table entries in the Business Process database. The Audit Log file grows in direct proportion to user activity and needs to be maintained on a regular basis.

To examine the contents of the Audit Log file

Use one of these commands:

```
cat filename
more filename
```

You can also open the file using an ASCII editor, such as *vi* or *emacs*.

To delete the contents of the Audit Log file, but not the file itself

Use this command:

```
cat /dev/null > filename
```

To delete the Audit Log file and its contents

Use this command:

```
rm filename
```

Note: Do not delete the Audit Log file unless the eLink Business Process Option is down. When the eLink Business Process Option is not running, deleting the log file has no effect on the system. When you restart the eLink Business Process Option, a new Audit Log file is created automatically.

Maintaining the Repository

The *`IcRepCheck`* and *`IcRepManage`* repository utilities – valid for both NFS and Repository Server repositories – let you manage unused content files and ensure the integrity of the InConcert database:

- *`IcRepCheck`* examines repository information in the database and compares it to the actual repository contents and vice-versa.

Note: *`IcRepCheck`* lists files that have no document objects in the database. Run this utility when unexpected errors occur, such as after restoring the system from backup.

- *`IcRepManage`* lists and removes content files with no references in the database – including all versions of documents other than the latest version. *`IcRepManage`* lists and removes files for documents that are not being referenced in the database. Run this utility routinely to clean up unused content files.

Notes: If you want old versions (versions other than the latest version) of your documents, do not run *`IcRepManage`*. *`IcRepManage`* will delete all the old versions of your documents.

All write activity involving repository directories should be performed through Business Process Option. Writing directly to a repository location (without Business Process Option) can result in the loss or corruption of content files.

Using IcRepCheck to Check Repository Information

To check the condition of a repository, use the *`IcRepCheck`* utility. *`IcRepCheck`* lets you perform the following checks:

- Verify all paths from each repository content file back to the Business Process Option database.
- Verify the path from a specified Business Process document to the corresponding repository content file.
- Check for the existence of a particular content file specified by the user.
- List repository content files not referenced by the Business Process Option database.
- List all Business Process Option documents that contain no reference to any content file.

You can run the *`IcRepCheck`* utility to examine the integrity of existing repositories, repository content files, and information about repositories in the Business Process Option database. The Business Process Option must be running when using *`IcRepCheck`*.

Note: You can only run the *`IcRepCheck`* utility if you have Administrator privilege.

`IcRepCheck` has the following operating modes:

Interactive mode
Lets you examine repository information with a menu-based interface. Interactive mode lets you check all Business Process Option document objects for references to a specific repository content file and check all repository content files for references to a specific Business Process Option document object.

Command mode
Bypasses the menu interface and lets you check repository-to-database associations with a series of command line options. It does not provide all of the capabilities of the menu interface.

Note: The Business Process Option server must be running when using *IcRepCheck*.

Using IcRepCheck in Interactive mode

During Interactive mode, you specify the kind of check you want to make, such as: check repository, check a single file in a repository, or check a single document in the database. You must respond to a series of menus to further qualify the operation. Every menu includes a Quit option that returns you to the previous menu; selecting Quit from the main menu exits *`IcRepCheck`*.

Verifying Server Information

To verify server information before running *`IcRepCheck`*

1. Log into the Business Process Option on the machine hosting the server. Be sure you are logged in as a user that has write access to the repository directory (for example, *`icadmin`*). *`IcRepCheck`* behaves differently, depending on the user who starts it.

2. Be sure your `$IC_CLIENT` environment variable is set to:
   `icdba/password@server`

3. Go on to "Running IcRepCheck".

Running IcRepCheck

To run `IcRepCheck` in interactive mode:

```
IcRepCheck -icclient user/password@server
```

1. The *IcRepCheck* main menu is displayed:

```
Main Menu
1. Check Integrity of Repository
2. Check Integrity of Individual Content File
3. Check Integrity of Individual Document
4. Quit
```

2. To check all files in one or more repositories, choose Check Integrity of Repository. To check a specific content file, choose Check Integrity of Individual Content File. To check a specific document in the database, choose Check Integrity of Individual Document.
3. *`IcRepCheck`* displays the first of a series of submenus where you specify one or more of the following:
   - Repository
   - Kind of check such as content file against database or database against repository or file
   - Kind of file or document listing (long or short).
4. After making your selections, *`IcRepCheck`* displays the results of the check and returns to the latest menu. You can make another selection in that menu, or select Quit to return to the previous menu.

Using IcRepCheck in Command Mode

The command interface for *`IcRepCheck`* lets you check repository-to-database associations with a series of command line options. It does not provide all of the capabilities of the menu interface.

Syntax

```
IcRepCheck [ -icclient client_data ] [ -ictimeout seconds ]
```

```
IcRepCheck -icclient client_data [ -ictimeout seconds ]
{ -cont | -dbms } { -ll | -ls } [ -rep repository_pathname ]
```

Parameters

`-icclient` *`client_data`*

Client data in the form:

```
user/password@server
```

See Chapter 3, “Using the Batch Registry,” for details on this parameter.

`-ictimeout` *`seconds`*

Timeout interval, in seconds. The default timeout is 120 seconds.

`-cont | -dbms`

These two parameters determine the kind of check made by *IcRepCheck*:

`-cont`
: Checks for repository content files that are not referenced in the Business Process Option database.

`-dbms`
: Checks for Business Process Option documents in the database that are not associated with any repository content file.

`-ls`
: Produces a short listing (file name only) of content files. With `-cont`, the listing includes content files which are not referenced in the database. With `-dbms`, the listing includes content files that are referenced in the database but do not exist.

`-ll`
: Produces a long listing (file name, date, time stamp) of content files. With `-cont`, the listing includes content files that are not referenced in the database. With `-dbms`, the listing includes content files that are referenced in the database but do not exist.

Note: When multiple versions of a document are listed, all but the most recent version include the following information: the associated Business Process Option process and task, and name of the user who created the document.

`-rep` *`repository_pathname`*
: Checks only the repository specified by *`repository_pathname`*. If you omit this parameter, all repository content files are examined.

Using IcRepManage to Delete Unused Files from Repositories

The *`IcRepManage`* utility lets you list or remove document content files in a repository that are not referenced by the Business Process Option database. These files are considered “unused” files. The Business Process Option server must be running when using *`IcRepManage`*.

Note: The first time *`IcRepManage`* is run on a system that has had heavy PC GUI usage, it may take some time to complete this operation. You may wish to run it during off-hours. *`IcRepManage`* will output a series of dots while the operation is in progress. If *IcRepManage* fails to complete in the time allotted, you may need to increase the environmental variable `IC_TIMEOUT` to 99,999. This represents the number of seconds that *`IcRepManage`* will wait for a response from the server before timing out.

`IcRepManage` lets you:

- List all unused content files, in an individual repository or all repositories.
- Remove all unused content files, in an individual repository or all repositories.

In addition, *`IcRepManage`* deletes obsolete Task User Interface documents from the database and repository. Note that the TUI documents are not deleted unless the process they are related to has first been deleted by the process owner. To do this, open the "Completed Processes" folder in the Process Manager and choose **Process** > **Delete**.

Note: *`IcRepManage`* can be run only by users with Administrator privilege.

`IcRepManage` has an Interactive mode and Command mode. The interactive mode allows you to examine and remove files with a menu-based interface. The command mode bypasses the menu interface and lets you examine and remove files with command line options.

Upon completion, *`IcRepManage`* creates the log file `IcRepManage.log` in `$IC_HOME/logs`.

Using IcRepManage in Interactive Mode

In Interactive mode, you specify the repository (or repositories) you want to process and respond to a series of menus to further qualify the operation. Every menu includes a Quit option that returns you to the previous menu; selecting Quit from the main menu exits *`IcRepManage`*.

To process repositories in interactive mode:

1. At a shell prompt, enter the following command:

   ```
   IcRepManage
   ```

2. InConcert displays the *IcRepManage* main menu shown below.

   ```
   Main Menu
   
   1. Process All Repositories
   ```

```
2. Process Specific Repositories
3. Quit
```

3. To process all repositories, choose Process All Repositories. To process specific repositories, choose Process Specific Repositories.
4. *`IcRepManage`* displays the first of a series of submenus where you specify:
   - Type of operation you want to perform: list or remove
   - Type of file listing: long or short
5. After making your selections, *`IcRepManage`* either displays or removes the specified repository files, then returns to the latest menu. You can make another selection in that menu, or select Quit to return to the previous menu.

Using IcRepManage in Command Mode

The command interface for *`IcRepManage`* lets you examine or remove unused document content files with a series of command line options.

Syntax

```
IcRepManage [ -icclient client_data ] [ -ictimeout seconds ]
[ -ls | -ll ] [ -r ] [ repository_pathname ]
```

Parameters

`-icclient` *`client_data`*

The following items are client data in the form:

`user/password@server`

See Chapter 3, “Using the Batch Registry,” for details on this parameter.

`-ictimeout` *`seconds`*

Timeout interval, in seconds. The default timeout is 120 seconds.

`-ls | -ll`

These two parameters determine the kind of listing produced by *IcRepManage*:

`-ls`

Produces a short listing (file name only) of the unused content files in ***repository_pathname***. If you omit ***repository_pathname***, all unused content files in all repositories are listed.

`-ll`

Produces a long listing (file name, date, time stamp) of the unused content files in ***repository_pathname***. If you omit ***repository_pathname***, all unused content files in all repositories are listed.

`-r`

Removes unused content files from ***repository_pathname***.

`repository_pathname`

Specifies the repository to be examined by *IcRepManage*. If you omit this parameter, all repository content files are examined.

Optimizing the Business Process Option Database

You should execute the appropriate database optimization procedures on a weekly or monthly basis, or whenever there is a significant increase in the size of the Business Process database.

Moving Work Between eLink Business Process Option Servers

Using the `Batch Registry` utility, you can copy process definitions from one Process Option to another. This procedure lets you move work between eLink Business Process Option servers.

To copy a document or process from one server to another

1. In the source eLink Business Process Option server's `Batch Registry`, export the process with the Export Job command.
2. In the destination eLink Business Process Option server's `Batch Registry`, import the process with the `Import Job` command.

2 eLink Business Process Option Daemon Manager

The following sections describe how to use the Business Process Option Daemon Manager, its capabilities, and its parameters:

- About the Daemon Manager
- Restarting Failed Applications
- Setting Up the Daemon Manager Configuration File
- Setting Up an Application Under the Daemon Manager

About the Daemon Manager

The Daemon Manager (`IcCliMgr`) is an eLink Business Process Option Tuxedo server that allows you to start, stop, and restart registered daemon client applications that are running as Tuxedo Client Programs. You can use the Daemon Manager to start up and monitor vital daemons, such as the `IcEvtSched`, and restart them if they fail. It also provides the ability to configure and control how such applications will be restarted.

The Daemon Manager has its own configuration file, `IcCliMgr.cfg`, a text file that defines the applications to be supervised by the Daemon Manager and defines how the applications will behave during system runtime. `IcCliMgr.cfg` is located in `$IC_HOME/config`.

The Daemon Manager is booted when the Tuxedo domain and eLink Business Process Option are started up, and it is shut down as part of the shutdown process for the domain and eLink Business Process Option.

Restarting Failed Applications

Once the Daemon Manager is running, it monitors the daemon applications under its jurisdiction, and supervises the restarting of any applications that fail. Restart behavior is configured on a per-application basis. Each daemon application can run multiple concurrent instances (if the `BOOT` and `NCONC` parameters in the `IcCliMgr.cfg` file are each set to a value greater than 1).

Whenever a daemon process instance dies or whenever the domain is booted, the Daemon Manager attempts to start as many instances of the daemon application as necessary. It continues to start instances of the daemon application until either the number of active process instances matches the `BOOT` count, or the number of concurrent process instances equals the `NCONC` parameter. The number of concurrent instances of the daemon application cannot exceed the `NCONC` value.

Each daemon application managed by the Daemon Manager has a grace period associated with it (set by the `GRACE` parameter). If no value is assigned to the `GRACE` parameter in the `IcCliMgr.cfg` file, its default value is one hour (3600 seconds). When the Daemon Manager is booted for the first time, or when a process dies after a period of successful operation, the Daemon Manager begins measuring the elapsed time.

If a process instance dies before the `GRACE` period has elapsed, and the total number of retries is less than the `RETRY` parameter value, the Daemon Manager will attempt a retry. It continues to retry starting the process instance until it reaches the maximum number of retries during a `GRACE` period, which is set by the `FLUX` parameter. If the Daemon Manager cannot start the daemon application in `FLUX` number of attempts, it stops and waits the duration of the `GRACE` period before taking any further action.

When the GRACE period expires, the Daemon Manager will retry starting a process instance until the number of attempts exceeds the FLUX parameter, or the total number of retry attempts exceeds the RETRY parameter, or the daemon application is successfully started.

If a process instance dies after its GRACE period (which started when the process instance was last booted or restarted) expires, an attempt to launch a process instance is considered a *restart* rather than a *retry*. The RSTRT parameter sets the maximum number of restart attempts for the daemon application. By default, the NCONC, MAXST, RSTRT, and RETRY parameters are set to -1, which represents an unlimited value.

Setting Up the Daemon Manager Configuration File

The Daemon Manager configuration file, IcCliMgr.cfg, defines each Tuxedo daemon application to be controlled by the Daemon Manger by storing its name, current directory, the location of the daemon's executable, any arguments required by the daemon, the daemon application's environment variables, and information used to configure auto-restarting the daemon in case of failure.

The only mandatory configuration parameter assignment for a daemon application is PROGRAM. We recommend using at least the NAME and DIR parameters as well, for ease of maintenance

Note: For each daemon application in the Daemon Manager configuration file, eLink Business Process Option will assign a default value to any configuration parameter that does not have an explicit parameter assignment, *except* for the PROGRAM parameter.

Tuxedo Security and the Daemon Manager

The programs managed by the Daemon Manager are Tuxedo clients. By default, they perform tpinit authentication using the libiccustom shared library. Since these daemon applications are not interactive, the IC_TUXEDO_APPLICATION_PASSWORD, `IC_TUXEDO_USER_NAME`, and `IC_TUXEDO_USER_PASSWORD` environment variables must be specified either in the environment file or the environment parameters list in the `IcCliMgr.cfg` file.

Configuration File Syntax

For environment variables or parameter lists, you can use single quotation marks (`'`), double quotation marks ("), or curly braces (`{}`) in configuration statements. Curly braces can be nested to arbitrary levels. The pound sign (#) is the comment delimiter. White space is ignored.

Each daemon application section in the configuration file is separated from the other sections by a line that contains *only* dash characters, and which must contain at least two dashes (`--`).

Configuration File Syntax for Parameters

Use the following syntax to assign values to parameters:

```
PARAMETER=value
```

```
PARAMETERLIST: value1 value2
value3
value4
value5
```

Use either a single space or a newline followed by an indent on the next line as the delimiter between values in a parameter list.

Configuration File Syntax for Environment Variables

Use the following syntax for environment variable assignments, which include the `ENVS` entry in `IcCliMgr.cfg`, and the contents of the environment variables file pointed to by the `ENVFILE` parameter. Note that the Daemon Manager configuration file syntax is similar, but *not identical,* to the Tuxedo environment file syntax.

`VAR=value`	Assigns the given value to the named variable.

`VAR:=value`	Prepends the given value to any existing value for the named variable. The new value will be listed *before* any previously defined value. If necessary, separate this new value from the existing one by a colon.
`VAR=:value`	Appends the given value to any existing value for the named variable. The new value will be listed *after* any previously defined value. If necessary, separate this new value from the existing one by a colon.
`VAR=-value`	Assign the given value to the named variable *only if the variable is not currently defined* (does nothave a value assigned). If the named variable has already been defined, use the current value.
`VAR@`	Remove any current definitions (values) for the named variable.

Mandatory Client Application Entries

You must include an entry for the `IcEvtSched` daemon in the Daemon Manager configuration file. You must include an entry for the `IcRepDaemon` in the Daemon Manager configuration file.

Note: Make sure the owner of the `IcRepDaemon` process and the repository directory is `icadmin`.

Daemon Manager Configuration Parameters

The following sections list all possible configuration parameters for a daemon application running under the Daemon Manager. They are grouped by the functionality affected by the parameter.

Basic Parameters

The basic parameters for any client application to be managed by the Daemon Manager are listed in Table 2-1.

Table 2-1 Basic `IcCliMgr.cfg` Parameters

Parameter	Default Value	Description
`ARGS:`	None.	Lists any input arguments used by the client application. Note that the list is preceded by a colon and that the list may be continued on subsequent lines, providing those lines are indented.
`DIR`	$IC_HOME/tuxapp	Specifies the daemon application's current working directory, as well as the path used for other parameters with relative path name values. This parameter can be used with the `PROGRAM` parameter to provide the complete path name to the executable and to provide the `ENVFILE`, `INPUT`, `OUTPUT`, and `ERROR` parameters.
`ENVFILE`	None.	Specifies the path name of the environment file used by this client application. You can specify the full path or you can specify it relative to the path from `DIR`.
`NAME`	Final value in the `PROGRAM` path	Specifies a logical name for the daemon application. This value must be unique.
`PROGRAM`	None.	Specifies the name of the daemon application's executable. This value must be unique. *Mandatory line for each client application*.

You can use the parameters listed in Table 2-2 to control additional aspects of a daemon application:

Table 2-2 Additional IcCliMgrParameters

Parameter	Default	Description
`ENVS:`	None.	Defines and sets environment variables for the specified daemon application. This can be used when only one or two values are needed or when an `ENVFILE` is not desired. Note that the list is preceded by a colon and that the list may be continued on subsequent lines, providing those lines are indented. Environment variables assigned by the `ENVS` entry override any values set in the `ENVFILE`.
`ERROR`	Same as `OUTPUT`.	Specifies the full path name of an error file used by the specified daemon application.
`GROUP`	Value of `NAME`.	Groups related daemon applications. This allows you to start and stop multiple daemon with a single shutdown command. For example, you could create a group of polling agents.
`ID`	Next available number in sequence, starting with first ID value given in this file.	Assigns a unique ID number to a daemon application, which can be used to control the daemon's startup and shutdown.
`INPUT`	`/dev/null`	Specifies the full path name of an input file used by the specified daemon application.
`OUTPUT`	`/dev/null`	Specifies the full path name of an output file used by the specified daemon application.

Restart Parameters

Use the parameters listed in Table 2-3 to specify how and when the client application will be restarted, and when additional instances of the client application are spawned:

Table 2-3 Client Application Restart Parameters

Parameter	Default	Description
`BOOT`	1	Specifies the number of instances of the daemon application to keep active. This is the number of application instances booted when the Daemon Manager is first started, and the number of active application instances that the Daemon Manager will maintain by starting new instances if any instances die.
`FLUX`	3	The number of retries permitted within a single `GRACE` period. If the Daemon Manager cannot start the daemon application in `FLUX` number of attempts, it stops and waits the duration of the `GRACE` period. After the `GRACE` period expires, the Daemon Manager will retry starting the daemon application again, providing that the `RETRY` value has not been exceeded.
`GRACE`	3600 seconds (1 hour)	Specifies the time period (in seconds) during which the death of a daemon application instance will trigger a retry attempt rather than a restart.
`HIST`	20	Specifies the number of process history entries for the daemon application that the Daemon Manager should maintain. Note that the process history is stored in dynamic memory, and therefore it is purged whenever the Daemon Manager is terminated.
`MAXST`	Unlimited.	Specifies the maximum number of times the daemon application can be started for any reason, including system boot, restarting, retrying, and manual control from the system command-line. This prevents the Daemon Manager from endlessly restarting a daemon application which continuously fails after restarting due to some larger problem.
`NCONC`	Unlimited.	Specifies the maximum number of active instances of the specified daemon application that can exist at any given time.

Table 2-3 Client Application Restart Parameters (Continued)

Parameter	Default	Description
`RETRY`	Unlimited.	Specifies the maximum number of times the Daemon Manager tries to immediately start a failing daemon application, including retries after the `GRACE` period expires.
`RSTRT`	Unlimited.	Specifies the maximum number of times the daemon application can be restarted. This prevents situations where the daemon application has a problem that makes it fail after restarting, and yet the Daemon Manager continues to restart it.

Setting Up an Application Under the Daemon Manager

The Daemon Manager can only control daemon applications that act as Tuxedo clients.

Note: Before editing the Daemon Manager configuration file, shut down the Daemon Manager by entering:

```
tmshutdown -g SVRGRP -i SRVID
```

where *SVRGRP* and *SRVID* are the group name and server ID assigned to the Daemon Manager in the UBBCONFIG file.

To add a client application:

1. Edit the `IcCliMgr.cfg` file and add the following statements:

```
NAME=appname
DIR=path_of_executable
PROGRAM=name_of_executable
```

Only the `PROGRAM` statement is mandatory, but we recommend using either comments or the `NAME` and `DIR` statements for ease of maintenance.

2. Add any of the following statements, as appropriate for your daemon application:

```
GROUP=groupname
ENVFILE=envfile
ARGS: arg1 arg2 arg3
INPUT=input
OUTPUT=output
ERROR=error
ENVS: envar1=value envar2=value envar3=value
```

3. Add any of the following restart configuration parameters, as appropriate for your daemon application:

   `BOOT`

   `FLUX`

   `GRACE`

   `HIST`

   `MAXST`

   `NCONC`

   `RETRY`

   `RSTRT`

4. Make sure that each region of configuration statements pertaining to a specific daemon application is separated from the adjacent regions by a line solely consisting of at least two dashes (--). Make sure that no characters other than dashes are used on this line.

5. Save your changes. When you reboot the domain, the Daemon Manager will run the specified daemon application.

6. Alternatively, if you had shut down only the Daemon Manager and not the entire domain, you can enter the following to reboot the Daemon Manager:

   ```
   tmboot -g SVRGRP -i SRVID
   ```
   where *`SVRGRP`* and *`SRVID`* are the group name and server ID assigned to the Daemon Manager in the UBBCONFIG file.

7. Use the `tmadmin` utility to verify the number of active process instances.

3 Using the Batch Registry

The Batch Registry commands are used to perform administrative functions on users, pools, documents, processes, repositories, privileges, classes, triggers, events, and actions. Most of these functions are typically performed by the eLink Business Process administrator; others are performed by the Process Designer.

The Batch Registry utility may be run in one of the following two modes:

- **Interactive**. The Interactive mode uses a command window to accept commands and display responses.
- **Batch**. The Batch mode uses an input file containing a list of commands to run.

Note: Make sure that you installed the eLink Process Design Assistant. Otherwise, you cannot run the Batch Registry utility.

The following sections describe how to use the Batch Registry:

- Using Client Data in the Batch Registry
- The ictimeout and icclient Parameters
- Using the Batch Registry in Interactive Mode
- Using the Batch Registry in Batch Mode

Using Client Data in the Batch Registry

The eLink Business Process Option uses *client data* to pass information back and forth between servers and clients. Typically, this data is stored in the `IC_CLIENT` environment variable, and the value is in the following form:

```
user/password@server
```

where:

user	Name of the user to log in to the eLink Business Process server.
password	Password of *user*.
server	Logical name of the server to log in to (for example, Accounting).

This form is also used whenever client data needs to be passed explicitly (for example, when launching the Batch Registry from the command line).

The ictimeout and icclient Parameters

You can pass parameters to the Batch Registry in either Interactive or Batch mode. The *ictimeout* and *icclient* parameters are optional, but can always be passed. Other parameters may also be available. The following table describes the *ictimeout* and *icclient* parameters.

`-ictimeout` *`seconds`*	Optional. Specifies the timeout interval, in seconds, before the Batch Registry will stop trying to contact the eLink Business Process Engine. The default value is 120 (2 minutes).

`-icclient` *`client_data`*	Optional. Overrides all or part of the `IC_CLIENT` environment variable setting. `client_data` must be in the following form: `user/password@server`

Using the Batch Registry in Interactive Mode

Running the Batch Registry utility in Interactive mode lets you interactively issue Batch Registry commands. Interactive mode may be especially useful in performing one-time or non-routine tasks, such as creating a pool, deleting a user, or creating a subclass.

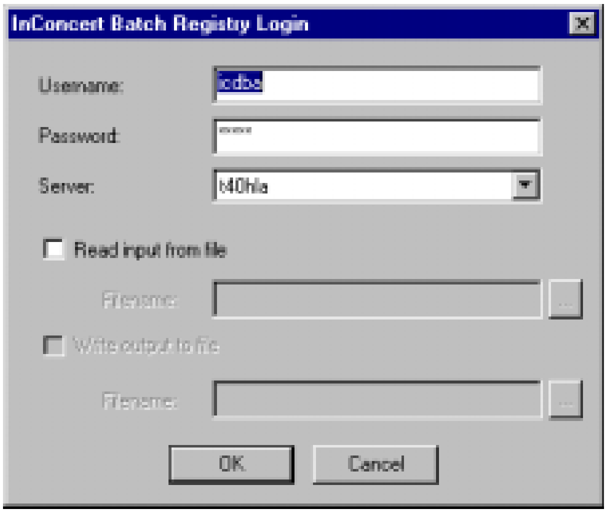
The PC and UNIX Batch Registry applications have different appearances, but they operate in the same way: they accept commands at a prompt and display the results.

To start the Batch Registry in interactive mode on a PC

1. Select **Start** ->**Programs** ->**BEA eLink** ->**Business Process Batch Registry** from the Windows NT Start Menu.

   The Batch Registry Login dialog box appears, shown in Figure 3-1.

Figure 3-1 The Batch Registry Login Dialog

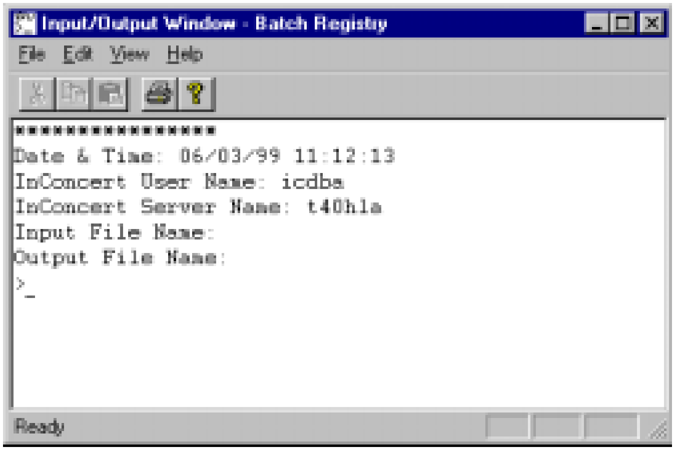


Username: icdba
Password: *****
Server: t40hla

2. Enter your user name, password, choose the server name, and click **OK**.

   The Batch Registry window appears, shown in Figure 3-2.

Figure 3-2 The Batch Registry Window



```
****************
Date & Time: 06/03/99 11:12:13
InConcert User Name: icdba
InConcert Server Name: t40hla
Input File Name:
Output File Name:
>_
```

3. Enter Batch Registry commands as desired.

To start the Batch Registry in interactive mode on UNIX

1. Log in as *icadmin*, or as any user with your *umask* set to `002`.
2. Make sure the IC-HOME directory is in your `PATH` environment variable, or change to `$IC_HOME`.
3. Enter the following command at the system prompt:

   ```
   IcBatchRegistry
   ```

   An informational message displays and the Batch Registry command prompt appears. For example:

   ```
   Host Name: Operations
   System User Name: icadmin
   Date & Time: 06/05/99 11:01:19
   InConcert User Name: icdba
   Process Engine Name: administration
   Input File Name:
   Output File Name:
   ```

Adding a Parameter to the Command Line

You can add the *`-icclient`* or *`-ictimeout`* parameters to the command line of the Batch Registry to override the `IC_CLIENT` environment variable.

Entering a Batch Registry Command in Interactive Mode

The procedure for entering a Batch Registry command on UNIX and PC platforms is identical. You enter a command at the prompt, specify parameters as necessary, and view the output.

To enter a Batch Registry command

1. Enter the command keyword and parameters using the syntax rules explained later in this chapter. You can continue the command on multiple lines by pressing **Return** after each line. The prompt changes to a double arrow (>>) after the first line.
2. End the command with a semicolon (;) and press **Return** to execute the command. For example:

   ```
   >list user;
   ```

Note: If you enter a command incorrectly or the command action fails, an error message displays.

Exiting the Batch Registry Utility

To exit the Batch Registry on a PC

Choose **File** > **Exit**.

To exit the Batch Registry on UNIX

Enter `quit` at the prompt.

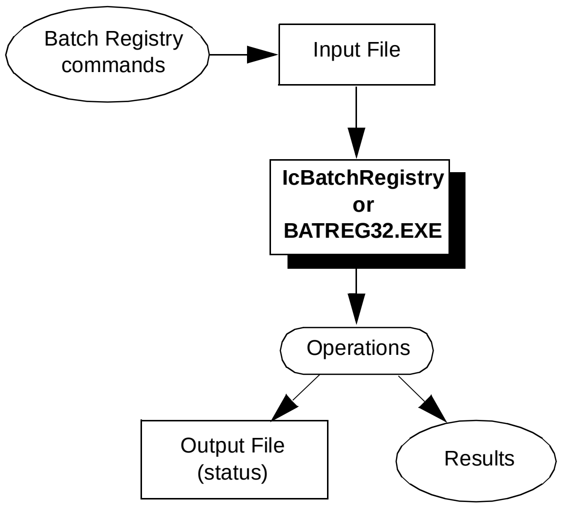
Using the Batch Registry in Batch Mode

When you run the Batch Registry in Batch mode, you specify the commands to be executed in a text file. Running the Batch Registry in Batch mode lets you:

- Execute and organize a large number of commands, check the syntax, and review the output.
- Run the same set of commands on a regular basis.
- Assemble and distribute command sets to be run by other eLink Business Process users.

The following figure shows how the Batch mode reads the commands from a text file, executes the commands one at a time, and writes the status of each operation to an output file.

Figure 3-3 How the Batch Registry Works in Batch Mode



To start the Batch Registry in Batch mode on a PC:

Open a command window and enter:

```
BATREG32 [-icclient client_data ] [ -ictimeout seconds ]
input_file [ output_file ]
```

To start the Batch Registry in Batch mode on UNIX:

From the command line, enter:

```
IcBatchRegistry [-icclient client_data ] [ -ictimeout seconds ]
input_file [ output_file ]
```

Command Line Parameters

If you do not specify the icclient parameter, the Batch Registry uses the information in the `IC_CLIENT` environment variable. The following table describes the *`input_file`* and *`output_file`* parameters.

`input_file`	**Required.** The input file parameter contains one or more of the Batch Registry commands.
`output_file`	**Optional.** If you specify this parameter, eLink Business Process appends the results of each batch operation to that file (the original contents are not overwritten). If you do not specify this parameter, eLink Business Process creates one with the name input_file.OUT.

Note: The *`input_file`* and *`output_file`* filenames cannot be the same.

Input File Format

The input file is a text file containing one or more of the Batch Registry commands as explained in Appendix A, “Batch Registry Command Reference,” You can create this file with any text editor.

An example of an input file for the Batch Registry is as follows:

```
LIST USER;
```

```
REMOVE USER "jones" FROM POOL "Writers";
DELETE USER "jones";
LIST POOL;
LIST USER;
```

The input file has the following format:

```
batch_command; [ batch_command; ] [ ... ]
```

You can enter several commands on the same line by separating the commands with a semi-colon. Spaces, tabs, and line terminators are ignored, and treated as white space, unless you use quotes.

Enclose string parameter values in single or double quotes using the quoting rules. Wildcards are permitted as indicated.

Output File Format

Output files have Header, Command, and Summary sections. Each section is separated by a linefeed character (LF).

Header Section

The Header section contains the following information:

- Name of the host on which the batch execution occurred
- Name of the user who ran the Batch mode
- Time the Batch mode was executed
- Server name for the BEA eLink Business Process Engine
- Name of the batch input file
- Name of the batch output file

Command Section

The command section contains the following information for each batch command:

- Batch command and associated parameters.
- Command output (if any).

```
LIST POOL
STATUS MESSAGE xxxxxxx
Writers
lee
jj
chris
Editors
luis
sri
kelly
```

Summary Section

The Summary section contains the following information:

- Number of successful batch commands
- Number of batch commands that produced an error

```
Success:   <# of commands>
Error:     <# of commands>
Total:     <# of commands>
```

A Batch Registry Command Reference

The following sections provide detailed explanations of the Batch Registry commands and command syntax:

- Batch Registry Command Syntax
- Overview of the Batch Registry Commands
- Command Reference

Batch Registry Command Syntax

Each Batch Registry utility command has syntax requirements. These syntax requirements are called character case, strings, and wildcards.

Using Character Case

Batch Registry utility commands have the following character case requirements:

- You can enter command keywords in uppercase, lowercase, or mixed-case characters.
- You must use the same case structure for parameter data values (for example, user names) as the case structure in which they were originally created.

- You must enter attribute names (for example, `PC_CACHE_DIR`) in uppercase characters.
- You must enter object or class names exactly as given. Internal class names are always capitalized; for example, "User."

To determine the case of a particular item, use the Batch Registry list commands, such as "List User" or "List Document".

Using Strings

You must enclose all character strings containing spaces, semi-colons, or quotes in single or double quotation marks. For example:

- "Insurance Job"
- 'Insurance Job'
- " Fred 'Davis' " (the single quotes become part of the name)
- " Fred Davis; "

In addition, the following strings are equivalent:

- accounting
- "ac"counting
- ac'count'ing
- "account"'ing'

Using Wildcards

Many commands let you use standard UNIX wildcards in file and document names, including asterisk (*) and percent (%).

Other commands use the '%' character to signify "all similar objects in the database." For example, the following command lists all active processes and process definitions:

```
LIST JOB NAME %;
```

Individual command descriptions indicate when you can use the ‘%’ wildcard for this purpose.

Overview of the Batch Registry Commands

The following table provides a brief description of each Batch Registry utility command.

The Batch Registry utility records errors in a file called `batreg.log`.

Category	Command	Description
Users	Create User	Creates a user record in the eLink Business Process Option database.
	Create User with Pool	Creates a user record in the Business Process database, simultaneously creating a pool record for the user.
	Delete User	Deletes a user record from the eLink Business Process Option database.
	Update User	Changes the name, password, email address, or other attribute of an existing eLink Business Process Option user.
	List User	Lists information about eLink Business Process Option users.
Pools	Create Pool	Creates a new pool.
	Delete Pool	Deletes a pool from the database.
	Update Pool	Renames an existing pool.
	Add User to Pool	Adds an eLink Business Process Option user to an existing Pool.
	Remove User from Pool	Removes a particular user from a pool.

Category	Command	Description
	Remove All from Pool	Removes all users from a pool.
	List Pool	Lists information about all defined pools.
Documents	Create Document	Creates one or more eLink Business Process Option document objects and associates these objects with an initial content file.
	Export Document	Prepares a document for use on another server.
	Import Document	Copies a document object and the associated content file from one Process Engine to another.
	Delete Document	Deletes one or more document objects from the database.
	List Document	Lists information about one or more documents.
Processes	Export Job	Prepares a process definition and all related tasks, roles, placeholders, triggers, and task user interface documents for use on another server.
	Import Job	Copies a process definition and all related tasks, roles, placeholders, triggers, and task user interface documents from one Process Engine to another.
	Delete Job	Deletes a process definition from the database.
	List Job	Lists information about one or more process definitions or active processes.
Repository	Create Repository	Creates a new repository.
	Set Default Repository	Changes the default repository.

Category	Command	Description
	List Repository	Lists information about repositories.
Privileges	Grant Privilege	Provides one or more users with the privilege to manipulate a process, document, or trigger.
	Revoke Privilege	Revokes privileges on processes, documents, and triggers.
	List Privilege	Lists process, document, or trigger privileges.
Class	Create Subclass	Creates a subclass of an existing class.
	Delete Class	Deletes a class from the database.
	List Class Hierarchy	Lists all defined classes and subclasses.
	List Class	Lists information about an eLink Business Process Option class.
	Promote Class	Promotes a class within its branch, retaining relationships to classes derived from it.
	Promote Instances	Promotes the instances of a specified class so that they become instances of that class's parent.
	Update Class - Add Attribute	Adds an attribute to a class definition.
	Update Class - Demote Attribute	Demotes an attribute from a class to all of its subclasses.
	Update Class - Promote Attribute	Promotes an attribute from a class to its parent.
	Update Class - Remove Attribute	Removes an attribute from a class definition and from all of the target class's subclasses.

Category	Command	Description
	Update Class - Rename Attribute	Changes the name of an attribute in a class definition and in all subclasses of the target class.
	Update Class - Set Default	Sets the default value for a user-defined attribute in a particular class.
Triggers	Create Trigger	Creates a global object trigger or global event trigger.
	Delete Trigger	Deletes a trigger from the database.
	List Trigger	Lists information about global triggers with a particular event and action specification.
Events	Register Event	Registers a user-defined event type for use in global trigger event specification.
	Unregister Event	Removes a registered event type from the database.
	Signal Event	Signals a registered event.
	List Event	Lists all registered event types.
Action	Register Message Action	Registers a triggered action specification that puts a message on a specific tuxedo queue.
	Register RPC Action	Registers a user-written program for use in a global trigger action specification.
	Unregister Action	Removes a registered action from the database.
	List Message Action	Lists all registered MESSAGE actions.
	List RPC Action	Lists all registered RPC actions.

Command Reference

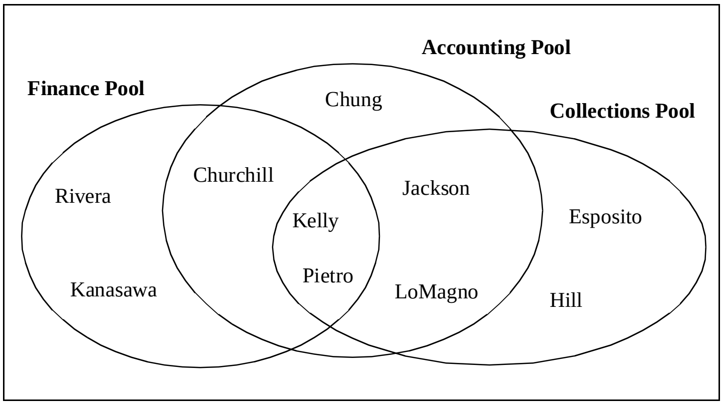
This section provides detailed descriptions of each of the Batch Registry commands.

Add User To Pool

Description		Adds an eLink Business Process Option user to an existing Pool.
Syntax		`ADD USER` *`user_name`* `TO POOL` *`pool_name`*`;`
Parameters	*`user_name`*	The name of an existing eLink Business Process Option user.
	`pool_name`	The name of the pool to which the user is added.
Example		`ADD USER Kelly TO POOL Finance;`

Note: A user must belong to a pool to perform an eLink Business Process Option task. Multiple eLink Business Process Option users can be added to the same pool and each user can belong to multiple pools. A typical installation has multiple pools and multiple users, as shown here.

Figure A-1 Typical Pool/User Organization



When you add users to a pool, they are granted the privileges of that pool while still retaining any pre-existing privileges.

Create Document File

Description		Creates one or more eLink Business Process Option document objects and associates these objects with an initial content file.
Syntax		`CREATE DOCUMENT file pathname [ NAME name_pattern ] [ KEYWORD keyword [,…keyword ] ] [ CLASS class_name ] [ [ ATTRIBUTE attribute_name VALUE attribute_value ] … ];`
Parameters	*pathname*	File(s) to be used as the initial document content; the file specification can include wildcards and full pathnames.
	name_pattern	Name of the document(s) to be created. The name pattern can include wildcards. If unspecified, the document names are the same as the file names.
	keyword	Keyword(s) for the new document(s); specify multiple keywords as a comma-separated string with no intervening spaces.
	class_name	The class for the new documents. *class_name* must be an existing subclass of Document. If unspecified, the documents become members of the Document class.
	attribute_name	Attribute to be set or updated; the attribute must be defined for *class_name* or the Document class.
	attribute_value	Value of *attribute_name*. The attribute value can be a string, integer, or date/time in the format: `mm/dd/yy hh:mm:ss`

Example

```
CREATE DOCUMENT FILE
/usr/local/smith/report/*.txt
NAME REPORT* KEYWORD september94 CLASS
Reports
ATTRIBUTE IC_APPLICATION VALUE "ascii
text";
```

Notes: A file is any information that can be stored, retrieved, or processed by the computer, such as an ASCII file, a scanned image, or word-processing file.

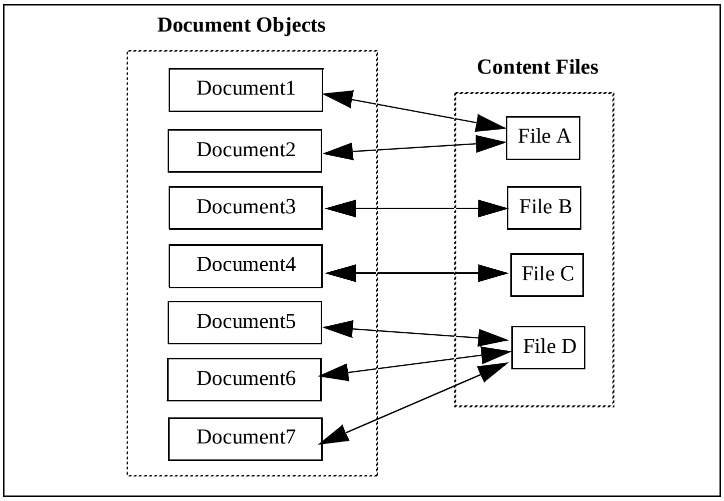
A file associated with an eLink Business Process Option document object is called a *content file*.

A document object is an ProductName database structure that describes one particular content file, such as the file name, creation date, and creator. Any file that you want to track or manage with ProductName *must* be associated with a document object.

One content file can be associated with multiple document objects. However, each object/file relationship is unique.

When you create a document, eLink Business Process Option copies the content file into the default repository.

Figure A-2 Document/Content File Associations



Document keywords describe the characteristics of the content file. For example, subject keywords might include annual report, product graphic, or insurance form. Keywords are used for document searches in the eLink Business Process Option application.

The optional *class_name* parameter let you categorize the document by class. By default, a new document becomes a member of the predefined Document class. You can use the *class_name* parameter to specify an existing subclass of Document; for example, Reports.

The ATTRIBUTE/VALUE keywords let you assign a value to a particular attribute for the class or override the default value; for example, writer name or document signoff date. You cannot create a new attribute, however; new class attributes can be defined only with “Update Class – Add Attribute”.

An important document attribute is called `IC_APPLICATION`. This predefined attribute specifies the kind of application used to create or update the document content file. The default value is “ascii text”, which means that the document is opened in whatever application you use to update ASCII files; for example, NotePad.

You map the value of `IC_APPLICATION` with an actual document-handling application using the setup program.

When you create a document, eLink Business Process Option grants you Update privilege on the document object; all members of the *admin* pool have Manage privilege on the document object; and all other users (PUBLIC) have Read privilege on the document object.

`Create Document` lists the following information for each new document:

- Content file name
- Document object ID

Create Pool

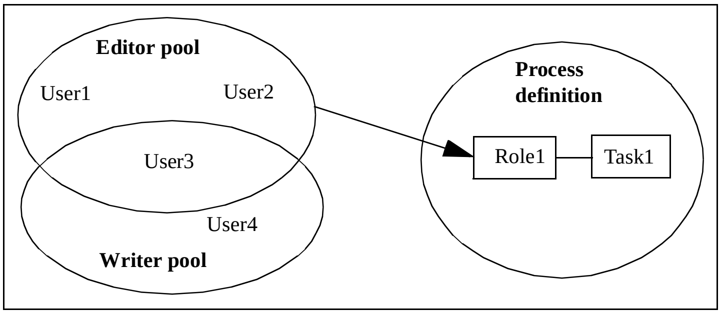
Description		Creates a new pool.
Syntax		`CREATE POOL` *`pool_name`* `[ CLASS` *`class_name`* `]` `[ [ ATTRIBUTE` *`attribute_name`* `VALUE` *`attribute_value`* `] … ];`
Parameters	*`pool_name`*	Name of the new pool.
	`class_name`	The class for the new pool. *`class_name`* must be an existing subclass of Pool. If unspecified, the pool becomes a member of the Pool class.
	`attribute_name`	Attribute to be set or updated; the attribute must be defined for *`class_name`* or the Pool class.
	`attribute_value`	Value of *`attribute_name`*. The attribute value can be a string, integer, or date/time in the format: `mm/dd/yy hh:mm:ss`
Example		`CREATE POOL Editors CLASS Writer;`

Notes: A pool is a group of users who perform the same job function. This command creates a pool record in the database. Once the pool record exists, you can populate the pool with “Add User To Pool”.

There is no restriction on the number of pools you can create or on pool organization. Typically, a site has multiple pools and individual users often belong to several pools at once.

You should create a pool for each eLink Business Process Option user. This step allows tasks to be assigned, delegated, or routed to individual users.

Figure A-3 Pool/Task Association



The optional `class_name` parameter lets you categorize the pool by class. By default, a new pool becomes a member of the predefined Pool class. You can use the `class_name` parameter to specify a user-defined subclass of Pool; for example, Writers.

The ATTRIBUTE/VALUE keyword lets you assign a value to a particular attribute for the class or override the default value. For example, security level, job description, or department name. You cannot create a new attribute, however; new class attributes can be defined only with “Update Class – Add Attribute”.

Create Repository

Description		Creates a new repository.
Syntax		`CREATE REPOSITORY location { location	hostname:port } PARTITION pathname TYPE type [CLASS class_name] [[ATTRIBUTE attribute_name VALUE attribute_value] ...] ;`
Parameters	*location*	Name of the repository host. For Repository Servers, *hostname* is the host in which the Repository Server is to run, and *port* is the port number in which the Repository Server is running. For NFS repositories, *location* is the name of the host on which the repository server is running.
	pathname	Full pathname of the repository directory.
	type	Repository type: 1 NFS 5 Repository Server
	class_name	Optional. The class for the new repository. The *class_name* must be an existing subclass of Repository. If unspecified, the repository becomes a member of the Repository class.
	attribute_name	Optional. Attribute to be set or updated; the attribute must be defined for *class_name* or the Repository class.
	attribute_value	Optional, but required if ATTRIBUTE is specified. Value of attribute_name. The attribute value can be a string, integer, or date/time in the format: `mm/dd/yy hh:mm:ss`
Example		`CREATE REPOSITORY LOCATION ServerA PARTITION /usr/local/ic/graphics_repos TYPE 1;`

Notes: A repository contains information about the directory used to store document content files, including the repository host name, directory pathname, and repository type. You can create multiple repositories to organize content files by, for example, file type, access privilege, or subject matter.

You must create the repository directory and start the respository server before running Create Repository (for example, with the UNIX *mkdir* command). Ensure that all eLink Business Process Option users have network access to this directory and can read and write files that are stored there.

If you change the default repository, existing documents remain in the original repository, including the system documents described above. As a result, you must keep all repositories accessible; otherwise, users may not be able to access old documents, their previously defined queries, or the system documents used to maintain folder persistence in the organizer windows.

The optional *class_name* parameter lets you categorize the repository by class. By default, a new repository becomes a member of the predefined Repository class. You can use the *class_name* parameter to specify a user-defined subclass of Repository; for example, Graphics.

The ATTRIBUTE/VALUE keywords let you assign a value to a particular attribute for the class or override the default value. For example, security level or kind of content file. You cannot create a new attribute, however; new class attributes can be defined only with “Update Class – Add Attribute”.

Create Subclass

Description		Creates a subclass of an existing class.
Syntax		`CREATE SUBCLASS` *`class_name`* `OF` *`parent_name`* `;`
Parameters	*`class_name`*	Name of the new class. The class name can include only the characters 0 through 9, a through z (upper and lower case), and underscore (_).
	`parent_name`	Name of an existing class. Be sure that you specify the class name with its original case structure. Base class names begin with an uppercase letter.

Example

```
CREATE SUBCLASS SPR OF Document;
UPDATE CLASS SPR ADD INTEGER ATTRIBUTE SEVERITY
DEFAULT 3;
UPDATE CLASS SPR ADD STRING ATTRIBUTE COMPONENT
DEFAULT unspecified;
```

Notes: A class is used to categorize ProductName objects by attributes. Attributes are inherited from the “parent” class, which is either one of the predefined base classes or a subclass of a base class. The base object classes include:

```
Binder
Document
Job
Link
Pool
Repository
Role
Task
Trigger
User
```

A Job class member is called a *process definition* and a Binder class member is called a *placeholder*.

Each parent class can have multiple subclasses and each subclass can be a parent of still other subclasses. You can list existing classes and subclasses with “List Class Hierarchy”.

Once you create a subclass, you can add attributes to the class definition with “Update Class – Add Attribute”.

Create Trigger

Description

Creates a global object trigger or global event trigger.

Syntax

```
CREATE TRIGGER [ NAME trigger_name ]
EVENT event_name ACTION action_specifier ;
```

An *action_specifier* can take these forms:

```
SENDMAIL POOL pool_name FORMAT mail_format [
text | file_name ]
```

```
JOBCREATE process_definition POOL owner_pool
NAME process_ID
```

```
PROCEDURE action_name ARGUMENTS parameter
```

```
MESSAGE action_name ARGUMENTS parameter
ATTRIBUTES a1 a2 a3
```

Parameters	*trigger_name*	Name of the new trigger. If unspecified, the action type becomes the trigger name: either `JOBCREATE`, `MESSAGE`, `SENDMAIL`, or `PROCEDURE`.
	event_name	Name of the event type that causes the trigger action to be performed.
	action_specifier	The action to be performed when the event occurs. This parameter can take any of the forms listed above for `JOBCREATE`, `MESSAGE`, `SENDMAIL`, and `PROCEDURE` actions.

Parameters for SENDMAIL action:

pool_name	Name of the pool to receive the mail message.
mail_format	Format of the mail message; either: ■ EVENT ■ FILE ■ FILE_AND_EVENT ■ STRING ■ STRING_AND_EVENT

Parameters for JOBCREATE action:

process_definition	Process definition id from which the active process is created.
owner_pool	Name of the process's owner pool; a member of this pool is granted Manage and Update privileges on the process.
process_name	Name of the active process; the active process name can be the same or different than the process definition name.

Parameters for PROCEDURE action:

action_name	Name of the registered action to send a message to when the trigger fires. Use "List Class" to list all registered actions.
parameter	Parameter(s) to be passed to the registered action; parameters must be enclosed in quotes. If the action takes no parameters, set this argument to null ("").

Parameters for MESSAGE action:

action_name	Name of the registered action to send a message to when the trigger fires. Use "List Class" to list all registered actions.
parameter	Parameter(s) to be passed to the registered action; parameters must be enclosed in quotes. If the action takes no parameters, set this to null ("").
a1, a2, a3	Attribute(s) of the object, such as *JobName* or *JobType*.

Examples

```
CREATE TRIGGER NAME new_job_trigger EVENT
JOB_COPY ACTION SENDMAIL
POOL mgrs FORMAT EVENT;
CREATE TRIGGER NAME shutdown_trigger EVENT
SERVER_SHUTDOWN ACTION SENDMAIL
POOL everyone FORMAT STRING "The Process Engine
has been shut down. Do not use InConcert until
further notice.";
CREATE TRIGGER NAME message_complete_trigger
EVENT TASK_COMPLETE MESSAGE notify_manager
ARGUMENTS parameter ATTRIBUTES TaskName;
```

Notes: A *trigger* defines an action to be performed automatically when a particular eLink Business Process Option event occurs, such as a completed process or task. There are three kinds of trigger:

- An *object trigger* must be explicitly associated with a particular InConcert object, such as a task. The trigger "fires" (performs the trigger action) when that object produces the specified event.
- A *global object trigger* is similar to an object trigger but is associated with *all* objects of a particular kind; for example, all tasks or all processes. In effect, a global object trigger is automatically associated with every object that can produce the same event.
- A *global event trigger* fires when certain special events occur, such as shutdown of the eLink Business Process Option server.

You can create object triggers only with InConcert's process design tools (for details, see the InConcert Process Designer's Guide). You can create global triggers only with "Create Trigger".

There are three kinds of trigger action:

- SENDMAIL — send a mail message to a user or pool
- CREATEJOB — create an active process
- MESSAGE — send a message to a named Tuxedo queue
- PROCEDURE — signal a user-written program

For a SENDMAIL action:

- The pool receiving the mail must exist.
- The mail message can take these forms:

EVENT – The message body has the following format:

```
(<date and time> GMT) <event name> by <username> - <text
description of event>
```

`STRING` – The message body is the *text* string in the action specification.

`STRING_AND_EVENT` – The message body is the *text* string plus the event message above.

`FILE` – The message body is the contents of the *file_name* in the action specification.

FILE_AND_ EVENT – The message body is the contents of *file_name* plus the event message above.

- Every mail message has the following subject line:

```
Process Engine <servername> Event Report
```

SENDMAIL triggers support any MHS-compliant mail package that runs under your PC/UNIX connectivity software

For a CREATEJOB action:

- Both the process definition and owner pool must exist.

For a PROCEDURE action:

- The action must have been registered previously with “Register RPC Action”.
- eLink Business Process Option repeatedly attempts to contact the trigger’s agent program (*action_name*) until it succeeds.

You automatically acquire Manage and Update privilege on any trigger you create. This means that you can delete the trigger and modify attribute values for the trigger’s class.

The following is a list of trigger actions:

BINDER_BIND_DOCUMENT	JOB_COMPLETE	TASK_ADD_RELATIVE
BINDER_DELETE	JOB_COPY	TASK_ADD_TASK
BINDER_UNBIND	JOB_CREATE_ACTIVE	TASK_COMPLETE
CLASS_ADD_ATTRIBUTE	JOB_CREATE_TEMPLATE	TASK_CONVERT
CLASS_CREATE_SUBCLASS	JOB_DELETE	TASK_COPY
CLASS_DEMOTE_ATTRIBUTE	JOB_RESUME	TASK_DELETE
CLASS_DESTROY	JOB_SUSPEND	TASK_DELETE_DEPENDENCY
CLASS_PROMOTE	LINK_COPY	TASK_ITERATE
CLASS_PROMOTE_ATTRIBUTE	LINK_CREATE	TASK_LATE_START
CLASS_PROMOTE_INSTANCES	LINK_DELETE	TASK_OVERDUE
CLASS_REMOVE_ATTRIBUTE	LINK_SET_DESTINATION	TASK_READY

CLASS_RENAME_ATTRIBUTE	POOL_ACQUIRE_FIRST_TASK	TASK_RELEASE
DOCUMENT_CANCEL_CHECK_OUT	POOL_ADD_USER	TASK_REPEAT
DOCUMENT_CHECK_IN	POOL_CREATE	TASK_SET_ITERATE_CONDITION
DOCUMENT_CHECK_OUT	POOL_DELETE	TASK_SET_PERFORM_CONDITION
DOCUMENT_COPY_VERSION	POOL_REMOVE_USER	TASK_SET_ROLE
DOCUMENT_CREATE	REVOKE_PRIVILEGE_POOL	TASK_SKIP
DOCUMENT_DELETE_VERSION	REVOKE_PRIVILEGE_PUBLIC	TASK_SUBTASKS_COMPLETE
DOCUMENT_DESTROY	REVOKE_PRIVILEGE_USER	TASK_TRANSFER
DOCUMENT_RESTORE_VERSION	ROLE_ASSIGN_POOL	TASK_WAITING
DOCUMENT_SET_CONTENT	ROLE_DEASSIGN	TRIGGER_CREATE
GRANT_PRIVILEGE_POOL	ROLE_DELETE	TRIGGER_DELETE
GRANT_PRIVILEGE_PUBLIC	SERVER_STARTUP	TRIGGER_FIRE
GRANT_PRIVILEGE_USER	TASK_ACQUIRE	USER_CREATE
JOB_ADD_BINDER	TASK_ACTIVATE	USER_DELETE
JOB_ADD_ROLE	TASK_ADD_DEPENDENCY	

Create User

Description Creates a user record in the eLink Business Process Option database.

Syntax

```
CREATE USER user_name ADDRESS
email_address [ WITH POOL ]
[ CLASS class_name ]
[ [ ATTRIBUTE attribute_name
VALUE attribute_value ] ... ] ;
```

Parameters

user_name	Name of the new user.
address	The new user’s email address.

WITH POOL	When this clause is present, a pool is created with the same name as the user. The user is added to this new pool.
`class_name`	The class for the new user. *class_name* must be an existing subclass of User. If unspecified, the user becomes a member of the User class.
`attribute_name`	Attribute to be set or updated; the attribute must be defined for *class_name* or the User class.
`attribute_value`	Value of *attribute_name*. The attribute value can be a string, integer, or date/time in the format: `mm/dd/yy hh:mm:ss`

Example

```
CREATE USER kelly ADDRESS
kelly@InConcert.COM CLASS
Manager
ATTRIBUTE CACHE_POLICY VALUE
"On Demand"
ATTRIBUTE PC_CACHE_DIR VALUE
C:\KELLY\CACHE_KE;
```

Notes: Each eLink Business Process Option user must have a user record in the eLink Business Process Option database in order to use eLink Business Process Option. A user record consists of a user name, an email address, and one or more attributes.

You must set the `CACHE_DIR` attribute for UNIX clients and may set `PC_CACHE_DIR` for PC clients. This attribute specifies the full pathname of the user's cache directory. If you do not set `PC_CACHE_DIR`, eLink Business Process Option uses a default cache directory location; `CACHE_DIR` has no default value.

You might also want to set the `CACHE_POLICY` attribute. This attribute can have the following values:

On Demand	Copies the document content file to the cache directory when the user explicitly opens or checks out an InConcert document. This is the default value for CACHE_POLICY.
On Task Start	Copies all document content files for a task to the cache directory when the user starts to work on the task, including read-only documents. This policy applies only to UNIX clients.

The optional *class_name* parameter lets you categorize the user by class. By default, a new user becomes a member of the predefined User class. You can use the *class_name* parameter to specify a user-defined subclass of User; for example, Manager.

The ATTRIBUTE/VALUE keywords allow you to assign a value to a particular attribute for the class or override the default value; for example, department name, job description, or years with the company. You cannot create a new attribute, however; new class attributes can be defined only with "Update Class – Add Attribute".

Delete Class

Description		Deletes a class from the database.

Syntax		`DELETE CLASS class_name;`
		`[{WITH | PROMOTE} SUBCLASSES]`
		`[PROMOTE INSTANCES];`

Parameters	*class_name*	Class to be deleted
Example		`DELETE CLASS NAME MyJobClass PROMOTE INSTANCES;`

Notes: The basic usage deletes a class with no subclasses or instances, and fails if any of these are present.

The `WITH SUBCLASSES` clause deletes the target class as well as its subclasses, provided that none have instances.

The `PROMOTE SUBCLASSES` clause promotes the target class's subclasses before deleting the target class, making the former subclasses direct subclasses of the target class's parent. Any attributes that these subclasses inherited from the target class become attributes of the (former) subclass. Any instances of the former subclasses are unchanged.

The `PROMOTE INSTANCES` clause promotes instances of the target class (and its subclasses, if the `WITH SUBCLASSES` clause is used), to become instances of the target class's parent. Any attributes inherited from the target class (or subclasses) are lost.

Note that use of the `PROMOTE INSTANCES` clause is the only way to delete a class that has instances. If the `PROMOTE INSTANCES` clause is not used, this command will not work (that is, if the class has instances, the command will not delete or promote subclasses).

Delete Document Name

Description		Deletes one or more document objects from the database.
Syntax		`DELETE DOCUMENT NAME` *`name_pattern`*`;`
Parameters	*`name_pattern`*	Name or pattern of the document(s) to be deleted.
Example		`DELETE DOCUMENT NAME Report*;`

Notes: A document object is an ProductName database structure that stores information about an associated content file. Delete Document deletes only the document object, leaving the content file intact. eLink Business Process Option classifies the content file associated with a deleted document object as an "unused content file."

This command deletes the document access control information, which removes any privilege relationships between the document and users and pools.

Delete Document lists the following information for each deleted document:

- Document name
- Document object ID

Delete Job Name

Description		Deletes a process definition from the database.
Syntax		`DELETE JOB NAME` *`name_pattern`*`;`
Parameters	*`name_pattern`*	Name or name pattern of the process definition(s) to be deleted.
Example		`DELETE JOB NAME AutoTheftClaim;`

Notes: A process definition describes a body of work that can be managed and tracked by eLink Business Process Option. When you delete a process definition, all information about the process is removed from the database.

This command deletes the process access control information, which removes any privilege relationships between the process definition and users and pools.

You cannot delete an active process. However, you can delete the associated process definition.

Delete Job lists the following information for each deleted process definition:

- Name of the deleted process definition
- Object ID of the deleted process definition

Delete Pool

Description		Deletes a pool from the database.
Syntax		`DELETE POOL` *`pool_name`*`;`
Parameters	*`pool_name`*	Name of the pool to be deleted.
Example		`DELETE POOL Editors;`

Notes: This command removes information about a pool from the database. The pool must be empty before you can delete it. To remove all users from the pool, use “Remove All From Pool”.

You cannot delete a pool if any process definition or instance includes a reference to the pool. You must wait until the active process(es) complete, and/or remove the pool references from all process definitions.

Delete Trigger

Description		Deletes a trigger from the database..
Syntax		`DELETE TRIGGER object_id;`
Parameters	*`object_id`*	Object ID of the trigger to be deleted.

Example		`DELETE TRIGGER`
		`000000050000115000010007000000C4;`

Notes: You cannot delete a trigger unless you have Manage privilege on the trigger object. Use “List Trigger” to obtain the trigger’s object ID.

Delete Trigger lists the following information:

- Trigger name associated with `object_id`
- Trigger object ID

Delete User

Description		Deletes a user record from the eLink Business Process Option database.
Syntax		`DELETE USER user_name;`
Parameters	*`user_name`*	Name of the user to be deleted.
Example		`DELETE USER Kelly;`

Notes: *`user_name`* cannot include wildcards.

This command removes all information about an eLink Business Process Option user from the database. Once you delete a user record, the user can no longer log in to eLink Business Process Option.

This command removes the user from all pools before deleting the user. If the user is currently working on one or more tasks, those tasks are released before the user is deleted.

Export Document

Description		Prepares a document for use on another server.
Syntax		`EXPORT DOCUMENT` *`document_id`* `TO` *`file_name_prefix`*`;`
Parameters	*`document_id`*	Object ID of the document to be exported; use "List Document" to obtain the object ID.
	`file_name_prefix`	Prefix for the output files produced during export. *`file_name_prefix`* can be a full path name but cannot include an extension; Export Document creates the extension automatically.
Example		**`EXPORT DOCUMENT 00000030000115000010007000000102 TO Doc1;`**

Notes: Copying an eLink Business Process Option document from one server to another is a two-step process:

1. Export the document with Export Document.
2. Import the document on the destination server with "Import Document".

Export Document prepares the document for transfer to the other server by creating two special output files:

- `file_name_prefix`.xdm
- `file_name_prefix`.con

"Import Document" uses these files as input when run on the destination server.

Export Job

Description		Prepares a process definition and all related tasks, roles, placeholders, triggers, and task user interface documents for use on another server.
Syntax		`EXPORT JOB job_id TO file_name_prefix ;`
Parameters	*`job_id`*	Object ID of the process definition to be exported.
	`file_name_prefix`	Prefix for the output file produced during export. *`file_name_prefix`* can be a full pathname but cannot include an extension; the extension is created automatically.
Example		`EXPORT JOB 000000030000115000010001000000091 TO JobA;`

Notes: Copying a process definition from one server to another is a two-step process:

Export the process definition with Export Job.

Import the process definition on the destination server with “Import Job”.

Export Job prepares the process definition for transfer to the other server by creating a special output file of the form *`file_name_prefix`*.xdm.

The Import Job command uses this file as input when run on the destination server.

In addition, for each task with a task user interface (created with the PC client), Export Job creates two more output files in the directory *`file_name_prefix.`*tui:

- *`document_id`*..xdm
- *`document_id`*.con

These files describe the task user interface in a portable format and are created by exporting the task user interface document. There is one such document for each task user interface in the process definition.

Grant Privilege

Description		Provides one or more users with the privilege to manipulate a process, document, or trigger.

Syntax		`GRANT PRIVILEGE` *`privilege`* `ON` *`object_id`* `TO subject ;`
		`subject can take these forms:`
		`USER` *`user_name`*
		`POOL` *`pool_name`*
		`PUBLIC`

Parameters	*`privilege`*	Privilege to be granted: Manage Update Copy Read
	`object_id`	Process, document, or trigger object ID. Use the List Job, List Document, or List Trigger command to obtain the object ID.
	`subject`	Subject to be granted privilege `USER`, `POOL`, or `PUBLIC`. This parameter can take any of the three forms listed above.
	`user_name`	Name of the user to be granted privilege.
	`pool_name`	Name of the pool to be granted privilege.

Example		`GRANT PRIVILEGE UPDATE ON 0000230000D09d000000002b00333300`
		`TO POOL Editors;`

Notes: You can grant privilege to a particular user, to a pool of users, or to all users (PUBLIC). There are four kinds of privilege:

- Manage
- Update

- Copy
- Read

Manage privilege lets you grant and revoke privilege on a process, document, or trigger, as well as delete a process, document, or trigger.

On a process, manage privilege lets you modify the process state (for example, suspend and resume the process).

Manage privilege does not extend Update, Copy, or Read privilege, but does allow you to grant Update, Copy, or Read privilege to yourself or other users.

The meaning of Update privilege varies for processes, documents, and triggers.

Update privilege on a process allows you to modify the objects comprising the process (for example, attach pools to roles or change the task structure). It does not extend Copy privilege to the subject.

Update privilege on a document allows you to checkout and checkin a document and modify its contents. It does not extend Read privilege to the subject.

Update privilege on a trigger allows you to modify the value of any attribute in the trigger's class.

Copy privilege applies only to process definitions (not active processes). Copy privilege allows you to create an active process from a process definition or copy a process definition to create a new one.

Read privilege applies only to documents. Read privilege allows you to view the document content file and create a copy of the document object.

The following table summarizes the privileges that can be granted to a user, pool, or PUBLIC for processes, documents, and triggers:

Process	Document	Trigger
Manage	Manage	Manage
Update	Update	Update
`Copy`	Read	N/A

Process	Document	Trigger
Manage	Manage	Manage
Update	Update	Update
Copy	Read	N/A

Privileges need to be granted only when the default privileges do not meet your requirements.

	Admin pool	User	Owner pool	PUBLIC
Base job template	Manage Update	—	—	Copy
New process definition	Manage	Manage Update	—	Copy
New active process	Manage	Manage Update	Manage Update	Copy
New document	Manage	Manage Update	—	Read
New trigger	Manage	Manage Update	—	—

eLink Business Process Option grants special privileges to members of two pools:

- The owner pool for a process
- The internal *admin* pool

Any pool that fills the predefined Owner role becomes the process owner pool. (The Owner role is associated with the process root task; this association cannot be removed. For more information, see the *InConcert Process Designer's Guide*.)

The Owner pool can be specified in the process definition or when the process is started. Members of the Owner pool receive Manage and Update privilege on the new active process. By default, members of the Owner pool also have the privilege to start the process because PUBLIC has Copy privilege on all process definitions.

Notes: If you want to restrict the users who can start a process, you must change the default privileges on process definitions.

Administrator privilege is a special privilege granted only to members of the internal *admin* pool. This privilege is not associated with any particular object but is needed to do the following:

- Grant and revoke privileges for other eLink Business Process Option users
- Update and delete users, pools, repositories, and classes

The *admin* pool always has at least one member, the eLink Business Process Option administrator (default name, 'icdba') who, for both practical and security purposes, cannot be removed from the pool; other users can be added to the *admin* pool with "Add User To Pool.

You should add eLink Business Process Option process designers to the admin pool. This is the only way they are permitted to create new attributes for the process definitions in use at your site.

Administrator privilege exists separately from the object-specific privileges described above and cannot be granted with Grant Privilege.

Import Document From

Description		Copies a document object and the associated content file from one ProductName server to another.
Syntax		`IMPORT DOCUMENT FROM` *`file_name_prefix`* `[ CLASS` *`class_name`* `] ;`
Parameters	*`file_name_prefix`*	File name prefix of the exported document, including the full path name if necessary. *`file_name_prefix`* must have been established previously with "Export Document".
	`class_name`	The class for the new document. *`class_name`* must be the Document class or one of its subclasses. If unspecified, eLink Business Process Option uses class matching to determine the class.

Example

```
IMPORT DOCUMENT FROM Doc1 CLASS
Document;
```

Notes: Import Document reads the output files produced by Export Document and creates a new document on the destination server. The exported document files must be readable by the Batch Registry user and include:

- `file_name_prefix`.xdm
- `file_name_prefix`.con

If you specify a class when you import the document, eLink Business Process Option sets the class attributes to their default value. If you do not specify a class when you import the document, eLink Business Process Option uses *class matching* to determine the document's class and attribute values. The imported document retains the class and attribute values of the original (exported) document when all of the following conditions are true:

- The original class name is defined on the destination server
- The class definition on the destination server includes the original class attribute names
- Corresponding attributes on each server are the same type (for example, string)

Class matching is concerned only with the exported document class name and attributes. Therefore, class matching can still succeed if a class on the destination server has additional (unexported) attributes. These additional attributes (if any) are set to their default value.

If eLink Business Process Option cannot determine the document class, the import operation fails. There are two ways that this can happen:

- You specify a class name during import with no match on the destination server.
- You do not specify a class name during import, but the original class has no match on the destination server.

eLink Business Process Option performs the following actions when a document is imported successfully:

- Creates a new document object; the imported document retains the document name and keyword(s) from the original (exported) document.

- Copies the document content file into the destination server's default repository; the document content file retains the name of the original (exported) document content file.
- Checks in the new document.

Import Document lists the following information about the imported document:

- Document name
- Document object ID

Import Job From

Description		Copies a process definition and all related tasks, roles, placeholders, triggers, and task user interface documents from one ProductName server to another.
Syntax		`IMPORT JOB FROM` *`file_name_prefix`* `[ CLASS` *`class_name`* `];`
Parameters	*`file_name_prefix`*	File name prefix of the exported process definition, including the full pathname if necessary. *`file_name_prefix`* must have been established previously with "Export Job".
	`class_name`	The class for the new Process Definition. class_name must be the Job class or one of its subclasses. If unspecified, eLink Business Process Option uses class matching to determine the class.
Example		`IMPORT JOB FROM JobA CLASS Job;`

Notes: Import Job reads the output files produced by Export Job and creates a new process definition on the destination server. The following files must be readable by the Batch Registry user:

- *file_name_prefix*.xdm (process definition export file)
- *document_id*.xdm, *document_id*.con (task user interface export files; one set per task)

An imported job has task user interface documents only if it was created on the PC client. The document export files are stored in the *file_name_prefix*.tui directory.

If you do not specify a class name when you run this command, eLink Business Process Option uses *class matching* to determine the class of the imported process definition and its related roles, placeholders, tasks, and triggers. These objects retain their original class and attribute values when all of the following conditions are true:

- The original class name is defined on the destination server
- The class definition on the destination server includes the original class attribute names
- Corresponding attributes on each server are the same type (for example, string)

If class matching fails, the imported object becomes a member of its base class (for example, Role) and its attributes are set to their default value.

Since class matching is concerned only with the exported class names and attribute names, class matching may still succeed if a class on the destination server has additional (unexported) attributes. These additional attributes (if any) are set to their default value.

If eLink Business Process Option cannot determine the class of the process definition itself, the import operation fails. There are two ways this can happen:

- You specify a class name during import with no match on the destination server.
- You do not specify a class name during import, but the original class has no match on the destination server.

eLink Business Process Option performs the following actions when a process definition is imported successfully:

- Creates a new process definition; the imported process definition retains the process name, role names, placeholder names, task names, task structure, dependencies, and triggers from the original (exported) process definition.

- Creates each task with the same due date, duration, priority, AutoActivate property, and PerformCondition as the corresponding task in the original (exported) process definition.
- For each task user interface in the original (exported) process definition, imports the task user interface document and creates a new task user interface document.
- Removes all document/placeholder assignments and any role/pool assignments with no corresponding pool on the destination server
- For each trigger in the original (exported) process definition, eLink Business Process Option creates a corresponding trigger in the new process definition if the original action specification can be matched:

For a `CREATEJOB` action, a match occurs only if the original trigger specifies a process definition name and pool name that exists on the destination server.

For a `SENDMAIL` action, a match occurs only if the original trigger specifies a pool name that exists on the destination server.

For a `PROCEDURE` action, a match occurs only if the original trigger specifies a registered action that exists on the destination server.

If an error occurs while the import operation is in progress, the process definition is created with a subset of the exported roles, placeholders, tasks, and triggers.

The PerformCondition for an imported task remains blank (unset) if it refers to a process or task attribute that is not defined on the destination server.

Import Job lists the following information about the new process definition:

- Object name and object ID.
- For each role — the role name, a status flag (`TRUE` or `FALSE`) indicating if class matching succeeded, and any information lost during import, such as the original role/pool assignment.
- For each placeholder — the placeholder name, a status flag (`TRUE` or `FALSE`) indicating if class matching succeeded, and any information lost during import, such as the original document/placeholder assignment.
- For each trigger — the trigger name, a status flag (`Yes` or `No`) indicating if the trigger was created (this field is No if the trigger's action

specification could not be matched), and another status flag (`TRUE` or `FALSE`) indicating if class matching succeeded.

- For each task — the task name and a status flag (`TRUE` or `FALSE`) indicating if class matching succeeded.

List Class

Description		Lists information about an eLink Business Process Option class.
Syntax		`LIST CLASS` *`class_name`*`;`
Parameters	*`class_name`*	Name of class to list information about.
Example		`LIST CLASS Application;`

Notes: Be sure to specify all eLink Business Process Option base class names with an initial cap; for example, Role, not ROLE or role.

See “Create Subclass” for more information about classes.

List Class lists the following information for each class attribute:

- Attribute name
- A flag indicating if the attribute is inherited: 0=False, 1=True
- Attribute data type: `STRING`, `INTEGER`, or `DATETIME`
- Default value

List Class Hierarchy

Description		Lists all defined classes and subclasses.
Syntax		`LIST CLASS HIERARCHY [WITH COUNT];`
Parameters	None.	
Example		`LIST CLASS HIERARCHY WITH COUNT;`

This function may take a few minutes to execute.

List Class Hierarchy lists the following information for each class:

- Name of the class
- Name of the class's parent class

When the `WITH COUNT` clause is present, the number of instances of each class is displayed along the class name. Instance counts are only for the indicated class and do not include subclasses of the indicated class.

List Document Name

Description		Lists information about one or more documents.
Syntax		`LIST DOCUMENT NAME` *`name_pattern`*`;`
Parameters	*`name_pattern`*	Document name or name pattern; *name_pattern* can include wildcards.
Example		`LIST DOCUMENT NAME Report%;`

List Document lists the following information for each document:

- Document name
- Document object ID

List Event

Description		Lists all registered event types.
Syntax		`LIST EVENT;`
Parameters	*`None.`*	
Example		`LIST EVENT;`

Notes: Registered event types are user-defined events that can cause a trigger to fire. They are registered by the Register Event command and are specified during trigger creation.

List Event lists the following information for each registered event type:

- Name of the event type
- # of the event type
- Event signalling interval
- Repeat interval (in seconds) for signalling the event

List Job Name

Description		Lists information about one or more process definitions or active processes.
Syntax		`LIST JOB NAME` *`name_pattern`*`;`
Parameters	*`name_pattern`*	Process name or name pattern; *name_pattern* can include wildcards (for, example, ‘%’).
Example		`LIST JOB NAME %;`

List Job lists the following information for each process:

- Name of the process

- Process object ID
- Process type: `TEMPLATE` (process definition) or `ACTIVE`

List Message Action

Description Lists all registered MESSAGE actions.

Syntax `LIST MESSAGE ACTION;`

Parameters None.

Example `LIST MESSAGE ACTION;`

Notes: Registered MESSAGE actions put a message on a named Tuxedo queue. They are registered by “Register Message Action” and specified during trigger creation with “Create Trigger”.

List Class lists the following information for each registered action:

- Name of the action
- Name of the queue space
- Name of the Tuxedo queue

List Pool

Description Lists information about all defined pools.

Syntax `LIST POOL;`

Parameters None.

Example `LIST POOL;`

List Message Action lists the following information for each pool:

- Name of the pool
- Name of pool members

List Privilege

Description		Lists process, document, or trigger privileges.
Syntax		`LIST PRIVILEGE ON` *`object_id`*`;`
Parameters	*object_id*	Object ID of the process, document, or trigger whose privileges are to be listed. Use the List Job, List Document, or List Trigger commands to obtain the object ID.
Example		`LIST PRIVILEGE ON 0000000200003039000000000100000400;`

List Privilege lists the following information for each privilege on the specified object:

- Privilege type: Manage, Update, Copy, or Read
- Subject: User, Pool, or PUBLIC
- User name, pool name, or blank (indicating PUBLIC)

List Repository

Description		Lists information about repositories.
Syntax		`LIST REPOSITORY;`
Parameters	None.	
Example		`LIST REPOSITORY;`

List Repository lists the following information for each repository:

- Repository object ID
- Name of the host where repository resides, and the port number if the repository type is Repository Server
- Full path name of repository
- Repository type: 1=NFS, 5=RepServer or the default repository.

List RPC Action

Description	Lists all registered RPC actions.
Syntax	`LIST RPC ACTION;`
Parameters	None.
Example	`LIST RPC ACTION;`

Notes: Registered actions are user-written programs that can appear in a global trigger action specification. They are registered by “Register RPC Action” and specified during trigger creation with “Create Trigger”.

List Class lists the following information for each registered action:

- Name of the action
- Name of host on which action is run
- Remote Procedure Call (RPC) number of the registered action
- RPC version number
- Procedure number of the registered action

List Trigger

Description		Lists information about global triggers with a particular event and action specification..
Syntax		`LIST TRIGGER EVENT` *`event_name`* `ACTIONTYPE` *`action_type`*`;`
Parameters	*`event_name`*	Event that causes the trigger to execute.
	`action_type`	Trigger action type: `JOBCREATE`, `SENDMAIL`, `MESSAGE`, or `PROCEDURE`.
Example		`LIST TRIGGER EVENT job_copy ACTIONTYPE SENDMAIL;`

List Trigger lists the following information for each matching trigger:

- Trigger name
- Trigger object ID
- Name of the user who created the trigger
- Name of the event type that causes the trigger to fire
- Description of the trigger action, as follows:

`JOBCREATE`: process, pool, name

`SENDMAIL`: pool, format, type, text

`MESSAGE`: action name, arguments, attributes

`PROCEDURE`: action name, arguments

List User

Description		Lists information about eLink Business Process Option users.
Syntax		`LIST USER;`
Parameters	None.	
Example		`LIST USER;`

List User lists the following information for each user:

- login name
- email address
- value of `CACHE_DIR` attribute, indicating the full pathname of the user's cache directory
- value of `CACHE_POLICY` attribute ("On Demand" or "On Task Start")

Promote Class

Description		Promotes a class within its branch, retaining relationships to classes derived from it. Any attributes formerly inherited from classes between it and its new parent become attributes of the promoted class.
Syntax		`PROMOTE CLASS` *`class_name`* `[TO SUBCLASS OF` *`parent_name`*`];`
Parameters	*`class_name`*	Class to be moved in hierarchy.
	`parent_name`	Class that will be new parent of moved class; if you do no specify a parent, the class is promoted to be a sibling of its current parent.

Example		`PROMOTE CLASS MyDetailedJobClass TO SUBCLASS OF Job;`

Notes: This command is typically used during process design, when you decide that a class should have a more general usage than the usage originally implied by the class's position in the hierarchy.

Another use is to "preserve" a class, preparatory to deleting its current parent classThis command is used primarily to preserve instances prior to deleting a class. This functionality already exists in "Delete Class".

Promote Instances

Description		Promotes the instances of a specified class, so that they become instances of that class's parent. Any attributes defined by the former class are lost.
Syntax		`PROMOTE INSTANCES OF CLASS` *`class_name`*`;`
Parameters	*`class_name`*	Class to which instances are to be promoted.
Example		`PROMOTE INSTANCES OF CLASS MyJobClass;`

Quit

Description		Exits the Batch Registry.
Syntax		`QUIT;`
Parameters	None.	
Example		`QUIT;`

Notes: On UNIX, use `Quit` to exit the Batch Registry. Another way to exit is with the key sequence Ctrl + D.

On Windows, choose **File** > **Exit** to exit the Batch Registry.

Register Message Action

Description		Registers a triggered action specification that puts a message on a specific Tuxedo queue.
Syntax		`REGISTER MESSAGE ACTION` *`action_name`* `QUEUE` *`queue_name`* `QUEUESPACE` *`space_name`*
Parameters	*`action_name`*	Name of the action to be registered.
	`queue_name`	Name of message queue.
	`space_name`	Name of queue space.

Example		`REGISTER MESSAGE ACTION test QUEUE tuxedo_queue QUEUESPACE tuxedo_queue_space`
		
		`create trigger name tuxedo_queue event JOB_COPY action MESSAGE TestActionSpec "testing one two three" ATTRIBUTE ATTRIBUTE1 ATTRIBUTE2;`

Notes: A global trigger can place a message on a named Tuxedo message queue if it is registered with eLink Business Process Option as a *registered message action*. Registering an action makes eLink Business Process Option aware of the action's general operating characteristics.

Only users who belong to the *admin* pool can execute “Register Message Action”.

Do not use the same name for an RPC action and a MESSAGE action.

Register RPC Action

Description

Registers a user-written program for use in a global trigger action specification.

Syntax

```
REGISTER RPC ACTION action_name
ADDRESS agent_address PROG program#
VERSION rpc_version PROC procedure#
;
```

Parameters

action_name	Name of the action to be registered.
agent_address	Name of host on which action is run.
program#	Remote Procedure Call (RPC) program number for *agent_address*; *program#* must be an integer in the range allocated for user-defined RPC services.
rpc_version	RPC version; this argument typically is set to 1 but can be any integer representing the RPC version.
procedure#	Integer procedure number to call within *action_name*.

Example

```
REGISTER ACTION run_status ADDRESS
Bluefish
PROG 591751050 VERSION 1 PROC 1;

CREATE TRIGGER NAME status_trigger
EVENT job_complete ACTION PROCEDURE
run_status ARGUMENTS "";
```

Notes: A global trigger can execute a user-written program if it is registered with eLink Business Process Option as a *registered RPC action*. Registering an action makes eLink Business Process Option aware of the program's general operating characteristics. The program itself is usually written in C or C++

with embedded eLink Business Process Option API function calls. The program must conform with the guidelines for agent applications described in the ICAgent documentation.

Only users who belong to the *admin* pool can execute “Register RPC Action”.

Do not use the same name for an RPC action and a MESSAGE action.

Register Event

Description		Registers a user-defined event type for use in a global trigger event specification.
Syntax		`REGISTER EVENT` *`event_name`* `TYPE` *`event_#`*`;`
Parameters	*`event_name`*	Name of the event type to be registered; this name must be unique among registered event types.
	`event#`	Event type # of *event_name*; this integer must be unique among registered event types and must be greater than 10000.
Example		`REGISTER EVENT daily TYPE 10001;` `SIGNAL EVENT daily AT "06/07/99 16:30:00";` `CREATE TRIGGER NAME status_trigger EVENT daily ACTION JOBCREATE group_status POOL mgr ID group_status;`

Notes: A global trigger can fire on a user-defined event type if it is registered with eLink Business Process Option. Registered event types are usually time-based, and are signalled periodically using the timing information established with “Signal Event”.

Only users who belong to the *admin* pool can execute the Register Event command.

Remove All From Pool

Description		Removes all users from a pool..
Syntax		`REMOVE ALL FROM POOL` *`pool_name`* `;`
Parameters	*`pool_name`*	Pool whose members are to be removed.
Example		`REMOVE ALL FROM POOL Editors;`

Notes: Remove All From Pool has two uses:

- To prepare a pool for deletion (a pool must be empty before it can be deleted)
- To clear a pool of current users

This command does not delete the pool itself.

Remove User From Pool

Description		Removes a particular user from a pool.
Syntax		REMOVE USER *user_name* FROM POOL *pool_name* ;
Parameters	*`user_name`*	Name of the user to be removed.
	`pool_name`	Name of the pool from which the user is to be removed.
Example		REMOVE USER Kelly FROM POOL Editors;

Notes: You might need to remove a user from a pool for any of these reasons:

- User reassigned to another pool

- User deleted from database
- Pool is going to be deleted

This command does not remove the user record from the database; use “Delete User”.

Revoke Privilege

Description		Revokes privileges on processes, documents, and triggers.

Syntax		`REVOKE PRIVILEGE` *`privilege`* `ON` *`object_id`* `FROM` *`subject`*`;`
		subject can take these forms:
		`USER` *`user_name`*
		`POOL` *`pool_name`*
		`PUBLIC`

Parameters	*privilege*	Privilege to be revoked: ■ Manage ■ Update ■ Copy ■ Read
	object_id	Process, document, or trigger object ID. Use the List Job, List Document, or List Trigger commands to obtain the object ID.
	subject	Subject from which to remove privilege. This parameter can take any of the three forms listed above.
	user_name	Name of the user from which the privilege is to be removed.
	pool_name	Name of the pool from which the privilege is to be removed.
Example		`REVOKE PRIVILEGE UPDATE ON 00000002000030390000000100000400 FROM PUBLIC;`

Notes: You cannot revoke Manage privilege from the *admin* pool or any of its members.

You need to revoke privileges on a process, document, or trigger if the current settings exceed the requirements of your application.

Copy privilege applies only to process definitions. Read privilege applies only to documents.

Since you can only revoke one privilege at time, you may need to execute Revoke Privilege several times to remove all the privileges you want.

Set Default Repository

Description		Changes the default repository.
Syntax		`SET DEFAULT REPOSITORY` *`repository_id`*`;`
Parameters	*`repository_id`*	Object ID of the new default repository. Use “List Repository” to obtain the object ID.
Example		`SET DEFAULT REPOSITORY 000000020000303900000000B00000403;`

Notes: The *default repository* is typically the repository used most often by the majority of users. The software installation procedure establishes the initial default repository. You can change the default repository to another repository with the Set Default Repository command. The new default repository must have been created previously with “Create Repository”.

Once you change the default repository, it is used to store the document content files for *new* eLink Business Process Option documents. Existing documents, created when a different repository was the default, remain in the repository in which they were originally created, and can still be accessed from a task.

Signal Event

Description		Signals a registered event.
Syntax		SIGNAL EVENT *`event_name`* { NOW | CLEAR | AT *`date_time`* [REPEAT *`repeat_interval`*] };
Parameters	*`event_name`*	Name of the registered event to be signaled.
	`date_time`	Date and time for signalling the event. You must include both the date and the time. The *date_time* value must be entered in double quotes, in the format: `mm/dd/yy hh:mm:ss`
	`repeat_interval`	Repeat interval (in seconds) for signaling the event; *repeat_interval* must be greater than or equal to 3600.

Example		`SIGNAL EVENT weekly AT "06/07/99 12:00:00" REPEAT 604800;`
		`SIGNAL EVENT update NOW;`

Notes: This command establishes the timing information for a registered event – a user-defined event created with “Register Event”. Registered events must be signalled explicitly for an associated trigger to fire. There are three kinds of timing specification:

- `NOW` signals the event immediately.
- `CLEAR` stops signalling an event.
- `AT` signals the event at the specified date/time with an optional repeat interval.

Only users who belong to the *admin* pool can execute the Signal Event command.

Unregister Action

Description		Removes a registered action from the database.
Syntax		`UNREGISTER ACTION` *`action_name`*`;`
Parameters	*`action_name`*	Name of the action to be unregistered.
Example		`UNREGISTER ACTION run_status;`

Notes: Only users who belong to the *admin* pool can execute the Unregister Action command.

Unregister Action removes the value given by *action_name* from either the Action RPC Table or the Action Message Queue Table.

Unregister Event

Description		Removes a registered event type from the database.
Syntax		`UNREGISTER EVENT` *`event_name`*`;`
Parameters	*`event_name`*	Name of the event type to be unregistered.
Example		`UNREGISTER EVENT weekly;`

Notes: Only users who belong to the *admin* pool can execute the Unregister Event command.

Unregister Event lists the following information:

- Event type name
- Event type #

Update Class – Add Attribute

Description		Adds an attribute to a class definition.
Syntax		`UPDATE CLASS` *`class_name`* `ADD` *`data_type`* `ATTRIBUTE` *`attribute_name`* `DEFAULT` *`default`*`;`
Parameters	*`class_name`*	Name of the class to be updated.
	`data_type`	The attribute's data type: ■ STRING ■ INTEGER ■ DATETIME
	`attribute_name`	Name of the new attribute. Use only the characters 0 through 9 and A through Z (no lowercase characters).
	`default`	Default value for the new attribute.
Example		`UPDATE CLASS Appraisers ADD STRING ATTRIBUTE COUNTY DEFAULT MONROE;`

Notes: This command adds a user-defined attribute to an eLink Business Process Option base object class (for example, Role) or one of its subclasses. You can list existing classes with "List Class Hierarchy".

A subclass inherits the attributes of its parent; user-defined attributes are inherited like any other attribute. .

Update Class – Demote Attribute

Description

Demotes an attribute from a class to all of its subclasses. The attribute is removed from all instances of the target class. Instances of the target class’s subclasses are unchanged.

Syntax

```
UPDATE CLASS class_name
DEMOTE ATTRIBUTE attribute_name ;
```

Parameters

`class_name`	Name of the class to be updated.
`attribute_name`	Attribute to be demoted.

Example

```
UPDATE CLASS MyJobClass DEMOTE
ATTRIBUTE JobNumber;
```

Note: This command is typically used during workflow design, to rearrange attributes that were incorrectly assigned to a class.

Update Class – Promote Attribute

Description

Promotes an attribute from a class to its parent. The attribute becomes inherited by the target class, its siblings, and all their subclasses. Instances of the target class and its subclasses are unchanged. Instances of the target class’s siblings and their subclasses receive the default value of the attribute.

Syntax

```
UPDATE CLASS class_name
PROMOTE ATTRIBUTE attribute_name ;
```

Parameters

`class_name`	Name of the class to be modifed.
`attribute_name`	Attribute to be promoted.

Example		`UPDATE CLASS MyJobClass PROMOTE ATTRIBUTE JobNumber;`

Note: This command is typically used during workflow design, to rearrange attributes that were incorrectly assigned to a class.

Update Class – Remove Attribute

Description		Removes an attribute from a class definition, and by extension from all of the target class's subclasses. The attribute is removed from instances of the class and its subclasses.
Syntax		`UPDATE CLASS` *`class_name`* `REMOVE ATTRIBUTE` *`attribute_name`* `;`
Parameters	*`class_name`*	Name of the class to be updated.
	`attribute_name`	Attribute to be removed.
Example		`UPDATE CLASS MyJobClass REMOVE ATTRIBUTE JobNumber;`

Note: This command is typically used during workflow design, to rearrange attributes that were incorrectly assigned to a class.

Update Class – Rename Attribute

Description		Changes the name of an attribute in a class definition and, by extension, all subclasses of the target class. The attribute is renamed for all instances of the target class and any of its subclasses.
Syntax		`UPDATE CLASS` *`class_name`* `RENAME ATTRIBUTE` *`attribute_name`* `TO` *`new_name`*`;`
Parameters	*`class_name`*	Name of the class to be updated.
	`attribute_name`	Attribute to be renamed.
	`new_name`	New name of attribute.
Example		**`UPDATE CLASS MyJobClass RENAME ATTRIBUTE JobNumber TO JobNum;`**

Note: This command is typically used during workflow design.

Update Class – Set Default

Description		Sets the default value for a user-defined attribute in a particular class
Syntax		`UPDATE CLASS` *`class_name`* `SET ATTRIBUTE` *`attribute_name`* `DEFAULT` *`default;`*
Parameters	*`class_name`*	Name of the class to be updated.
	`attribute_name`	Name (in uppercase letters) of the attribute to be updated.

	default	Default value for `attribute_name`. The attribute value can be a string, integer, or date/time in the format: `mm/dd/yy hh:mm:ss`
Example		`UPDATE CLASS Appraisers SET ATTRIBUTE COUNTY DEFAULT WESTFORD;`

Notes: Use this command to modify only user-defined attributes. Modifying predefined attributes can sometimes confuse the eLink Business Process Option software.

Update Pool

Description		Renames an existing pool.
Syntax		`UPDATE POOL` *pool_name* `NAME` *new_name*`;`
Parameters	*pool_name*	Old pool name.
	new_name	New pool name.
Example		`UPDATE POOL Engineer NAME Test Engineer;`

Update User

Description

Changes the name, password, email address, or other attribute of an existing eLink Business Process Option user.

Syntax

```
UPDATE USER user_name [ NAME new_name ]
[ PASSWORD new_password ]
[ ADDRESS new_address ]
[ [ ATTRIBUTE attribute_name VALUE
attribute_value ] … ];
```

Parameters

user_name	Name of the user whose database record is to be updated.
new_name	User’s new name.
new_password	User’s new password.
new_address	User’s new email address.
attribute_name	Attribute to be set or updated; the attribute must be defined for the class in which the user was originally created.
attribute_value	Value of *attribute_name*. The attribute value can be a string, integer, or date/time in the format `mm/dd/yy hh:mm:ss`

Example

```
UPDATE USER Kelly NAME Kelly PASSWORD xyz
ADDRESS Kelly@InConcert.COM
ATTRIBUTE CACHE_POLICY VALUE "On Task
Start"
ATTRIBUTE PC_CACHE_DIR VALUE
C:\KELLY\CACHE_KE;
```

Notes: Two common uses for this command are to:

- Change the login name or password for a user
- Update the `PC_CACHE_DIR`, `CACHE_DIR`, or `CACHE_POLICY` attribute for a user

Index