Industry-specific

QAD SOLUTIONS

Manufacturing Applications

External Interface Guide
Q/LinQ

UNIX Installation
Windows Installation
Installation Reference
Using Q/LinQ
User Reference

78-0516C
MFG/PRO Versions 9.0, eB, eB2
September 2002

78-0516C

QAD Inc.
6450 Via Real
Carpinteria, California 93013
Phone (805) 684-6614
Fax (805) 684-1890
http://www.qad.com

Contents

About This Guide 1

Other MFG/PRO Documentation 2
Online Help 3
QAD Web Site 4
Conventions 4
Menu and Book References 4
Interface 5
Typographic 6

Chapter 1 Installation Overview 7

Installation Introduction 8
Audience 8
Installation Errata 8
Q/LinQ CD-ROM 8
Installations 8
Minimum Requirements 9
Installation Utility 10
Keyboard Commands 10
Log Files 11

Section 1 UNIX Installation 13

Chapter 2 Installation on UNIX. 15

Installation Overview 16
Mounting the CD-ROM 16
Running the Installation 17
Storing the CD-ROM 19
Modifying the PROPATH 19
Registering Q/LinQ (MFG/PRO eB2 Only) 20

Chapter 3 Communication and Mapping Setup on UNIX 21

Overview 22
Setting Up Non-MOM Communication 22
Setting Up MOM Communication 23
Install and Configure MQSeries MOM 24
Allocate Network Ports on the Q/LinQ Host Computer 26
Configure the MOM Client 27
Configure the Non-MFG/PRO Application 28
Starting and Stopping the MOM Adapter 28
Setting Up Mapping 29
Completing Setup 30

Section 2 Windows Installation 31

Chapter 4 Installation on Windows NT 33

Installation Overview 34
Running the File Server Installation 34
Running the Client Installation 37
Modifying the Client PROPATH (Optional) 39
Registering Q/LinQ (MFG/PRO eB2 Only) 40

Chapter 5 Communication and Mapping Setup on Windows 41

Overview 42
Setting Up Non-MOM Communication 42
Setting Up MOM Communication 43
Install and Configure MQSeries MOM 44
Allocate Network Ports on the Q/LinQ Host Computer 47
Configure the MOM Client 48
Starting and Stopping the MOM Adapter 49
Configure the Non-MFG/PRO Application 49
Setting Up Mapping 50
Completing Setup 51

Section 3 Installation Reference 53

Chapter 6 Running Q/LinQ on UNIX and Windows. 55

Running Q/LinQ in Mixed Operating-System Environments 56
Creating a Shared Initialization File 56

Chapter 7 Initialization Files and Installation Directories 59

Q/LinQ Initialization Files 60
Q/LinQ Initialization File 60
Q/LinQ MOM Adapter Initialization File 61
Q/LinQ Installation Directories 64
Version 64
Archive 65
Bin 65
Classes 65
Data 66
Explog 66
Implog 66
Mapspec 67
MOM 67

Scripts 67
Work 68
Lib 68
Samples 68

Chapter 8 UNIX System Upgrades 69

Section 4 Using Q/LinQ 71

Chapter 9 Q/LinQ Overview 73

Introduction 74
Q/LinQ Features 75
Q/LinQ Interface Architecture 76
Data Flow 78
Q/LinQ Menu 79

Chapter 10 Q/LinQ Data Exchange 81

Interface Message Standards 82
Asynchronous Interface 82
BOOI Messages 82
Multi-Tier Environments and MOM 83
Transaction Document Groups 84
Document Group Types 84
Document Group Assignments 85
Group Processing Example 86
Autoacknowledgments and Notifications 87
Autoacknowledgment Messages 88
Autoacknowledgment Generation 88
Acknowledgment Processing 89
E-Mail Notification 90
Exchanging Documents 90
Document Exchange with Text Files 90
Document Exchange with APIs 91

Q/LinQ APIs 92
Publishing API 92
Processing API 93
Mapping API 96
Communication APIs 99

Chapter 11 Setting Up Q/LinQ 103

Overview 104
Reviewing the Initialization File 104
Defining Control Program Values 104
Registering External Applications 107
Identify Application 107
Interface Control Parameters 108
Data Mapping Parameters 109
Communication Parameters 109
Miscellaneous Defaults 111
Return Codes and Messages 112
Registering Multiple Applications 112
Setting Up Document Specifications 115
Document Identification 115
Trading Partners and MOM Channels or Queues 116
Defining Code Mappings 116
Starting and Restarting Q/LinQ 118
Normal Start and Restart 118
Restart After System Failure 118
Starting and Stopping the MOM Adapter 119
Starting the MOM Adapter 119
Stopping the MOM Adapter 120
Preserving Data if a Connection Breaks 121
Testing Communication 122
Using the C Test Programs 122
Using the Java Test Programs 124

Chapter 12 Managing Documents 125

Processing Stage and Error Status 126
Assigning Processing Stages 126
Reviewing Document Processing Stage and Error Status 127
Document Groups and Error Recovery 128
Exporting Data 129
Exporting to External Applications 130
Forwarding to MFG/PRO 131
Send Export Document Sessions 131
Using Send Export Documents 132
Importing, Mapping, and Processing Data 136
Importing from External Applications 138
Receiving Import Documents 139
Mapping and Processing Import Documents 142
Repositioning the Command Window 144
Monitoring Q/LinQ Sessions 145
Editing Documents, Data, Control Tags 146
Editing CIM Documents 146
Editing Document Data 147
Editing Document Control Tags 148
Dumping Data to a File 150
Generating Reports 151
Deleting or Archiving Documents 153

Section 5 User Reference 155

Appendix A OAGIS BOD Cross-Reference 157

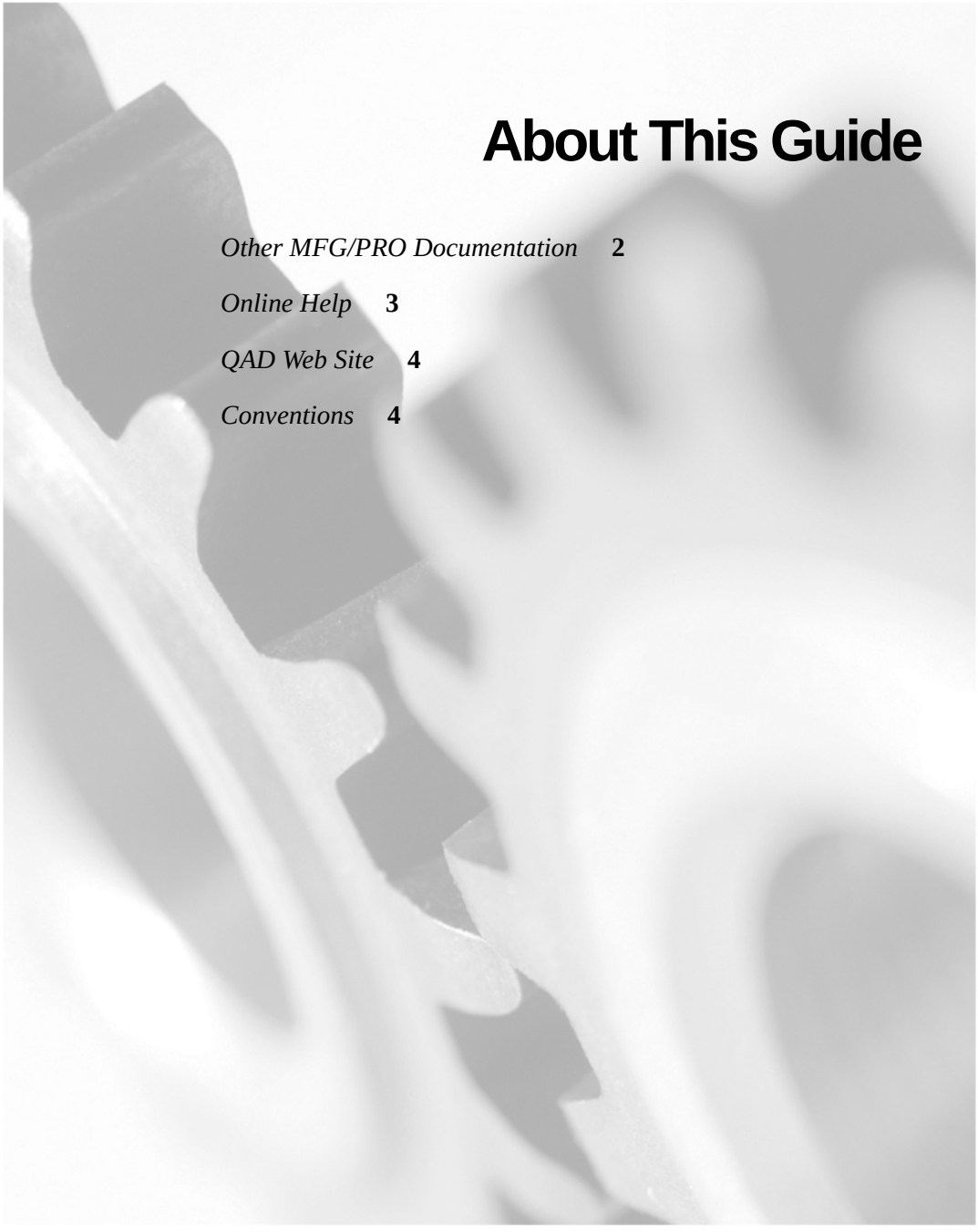
Overview 158
Conventions 158
BOD Stylesheet Specifications 159
Control Area Fields Cross-Reference Tables 159
Q/LinQ CONFIRM_BOD BODs 162
CONFIRM_BOD Example 165

Appendix B Text File Control Tags 167
Introduction 168
Q/LinQ Tags 168
ADI Tags 173
Application Control Tags 173

Appendix C Entity Diagrams 175
Q/LinQ I 176
Q/LinQ II 177
Q/LinQ III 178

Glossary 179

Index 185



About This Guide

Other MFG/PRO Documentation **2**

Online Help **3**

QAD Web Site **4**

Conventions **4**

This guide covers the installation, setup, features, and use of Q/LinQ for multiple versions of MFG/PRO; see the title page for a complete list.

Use this guide in conjunction with *Technical Reference: Q/LinQ*.

The first chapter is an introduction to the installation processes. The glossary and index are at the end of the guide following the appendices.

The body of the guide is organized in five sections:

- Section 1, “UNIX Installation,” beginning on page 13

  This section covers installations for UNIX environments as well as installation and setup for mapping and communication with external applications.

- Section 2, “Windows Installation,” beginning on page 31

  This section covers installations for Windows environments as well as installation and setup for mapping and communication with external applications.

- Section 3, “Installation Reference,” beginning on page 53

  This section covers setting up Q/LinQ to run in a mixed UNIX and Windows environment as well as the initialization files and installation directories.

- Section 4, “Using Q/LinQ,” beginning on page 71

  This section covers the asynchronous processes Q/LinQ uses to exchange documents with external applications, setting up Q/LinQ for data exchange, and managing the import and export of documents between Q/LinQ and external applications.

- Section 5, “User Reference,” beginning on page 155

  This section includes reference material on the Q/LinQ business object documents (BODs), text file control tags, and entity diagrams.

Other MFG/PRO Documentation

- For MFG/PRO installation instructions, refer to the appropriate installation guide for your system.
- For instructions on navigating the Windows and character environments:
  - For 9.0, refer to the *User Interface Guide*.

  - For MFG/PRO eB and eB2, see *User Guide Volume 1: Introduction.*
- For instructions on navigating the Network User Interface (NetUI) and Desktop environments:
  - For 9.0, refer to *Network User Interface Guide* or *User Guide: eB Desktop*.
  - For MFG/PRO eB, see *User Guide: eB Desktop and Network User Interface.*
  - For MFG/PRO eB2, see *User Guide: QAD Desktop*.
- For information on using MFG/PRO, refer to the *User Guides*. CIM Interface information changes by MFG/PRO version:
  - For 9.0, see *User Guide Volume 11: Manager Functions*.
  - For MFG/PRO eB and MFG/PRO eB2, see *User Guide Volume 9: Manager Functions*.
- To view documents online in PDF format, see *Documents on CD* and *Supplemental Documents on CD*.

Note MFG/PRO installation guides are not included on a CD. Printed copies are packaged with your software. Electronic copies of the latest versions are available on the QAD Web site.

Online Help

MFG/PRO has an extensive online help system. Help is available for most fields found on a screen. Procedure help is available for most programs that update the database. Most inquiries, reports, and browses do not have procedure help.

For information on using the help system for the different MFG/PRO interfaces, refer to the appropriate chapters of the books referenced in the previous section.

QAD Web Site

QAD's Web site provides a wide variety of information about the company and its products. You can access the Web site at:

http://www.qad.com

For MFG/PRO users with a QAD Web account, product documentation is available for viewing or downloading at:

http://support.qad.com/documentation/

You can register for a QAD Web account by accessing the Web site and clicking the Accounts link at the top of the screen. Your customer ID number is required. Access to certain areas is dependent on the type of agreement you have with QAD.

Most user documentation is available in two formats:

- Portable document format (PDF). PDF files can be downloaded from the QAD Web site to your computer. You can view them with the free Acrobat Reader. A link for downloading this program is also available on the QAD Web site.
- HTML. You can view user documentation through your Web browser. The documents include search tools for easily locating topics of interest.

Features also include an online solution database to help MFG/PRO users answer questions about setting up and using the product. Additionally, the QAD Web site has information about training classes and other services that can help you learn about MFG/PRO.

Conventions

Menu and Book References

This guide applies to multiple versions of MFG/PRO. A number of menus have been reorganized during these various releases. Differences in menu numbers are noted, when necessary, using the following convention:

Country Code Maintenance (2.14.1; 2.13.3.1 in eB)

The initial menu number identifies the program in the most recent release. The second menu number applies to the release specified and all earlier releases.

For Q/LinQ, the Version 9.0 menus are different from those for all of the other versions. For example, in 9.0, Register External Application is 36.8.1, while it is 36.8.1.1 in all of the other versions. The differences for Q/LinQ menu numbers are noted using this format:

Register External Application (36.8.1.1; 36.8.1 in 9.0)

In addition, some book titles have changed. References to these books use the title from the most recent release.

Interface

The software is available in several interfaces, depending on the release-level of MFG/PRO: Desktop or NetUI (Web browser), Windows, and character. To standardize presentation, the documentation uses the following conventions:

- MFG/PRO screen captures show the Windows interface.
- References to keyboard commands are generic. For example, choose Go refers to:
  - The forward arrow in Desktop
  - F2 in the Windows interface
  - F1 in the character interface

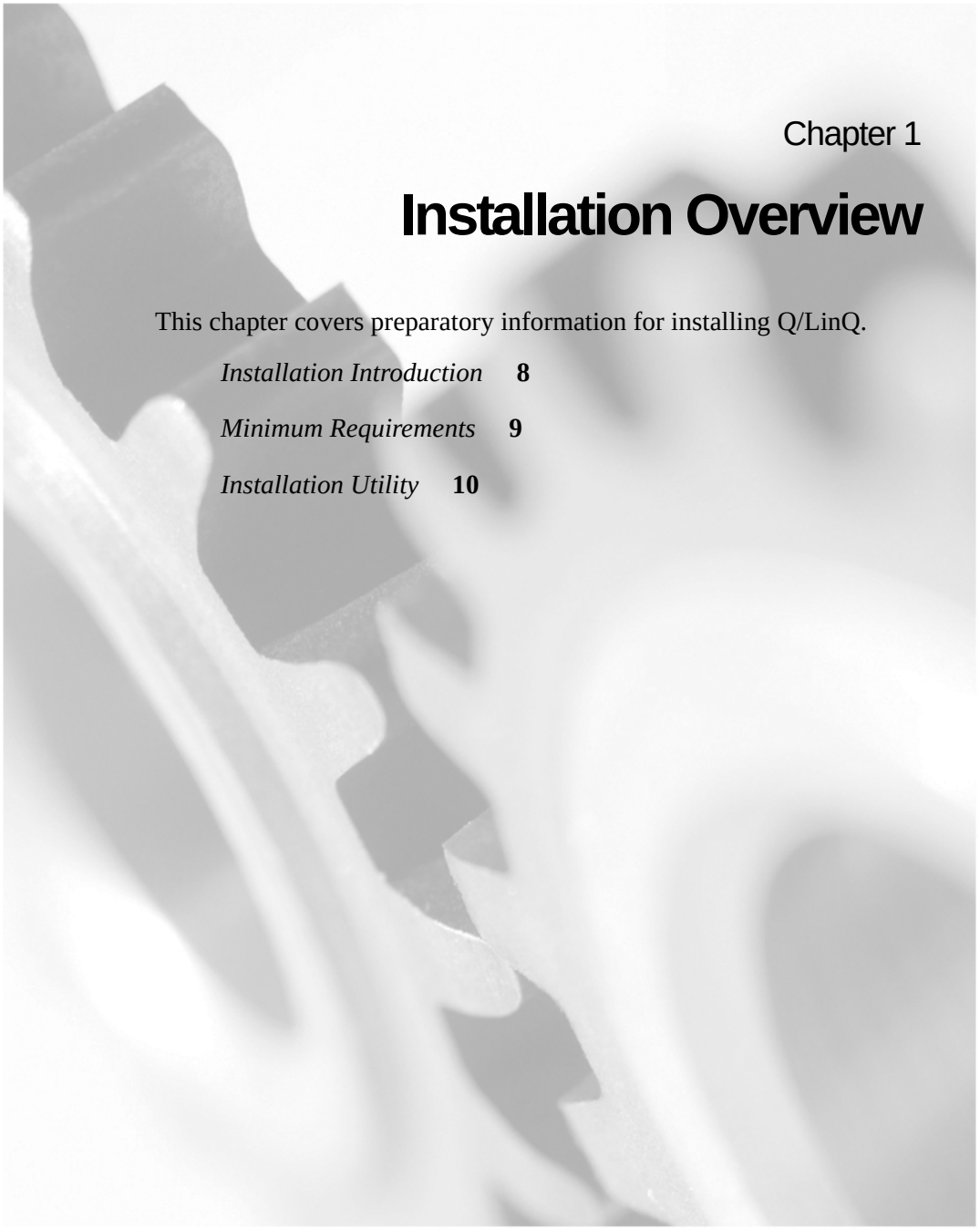
  In the character and Windows interfaces, the Progress status line at the bottom of a program window lists the main UI-specific keyboard commands used in that program. In Desktop, alternate commands are listed in the right-click context menu.

  For complete keyboard command summaries for each MFG/PRO interface, refer to the appropriate chapters of the books listed in “Other MFG/PRO Documentation” on page 2.

Typographic

This document uses the text or typographic conventions listed in the following table.

If you see:	It means:
`monospaced text`	A command or file name.
`italicized monospaced text`	A variable name for a value you enter as part of an operating system command; for example, *`YourCDROMDir`*.
`indented command line`	A long command that you enter as one line, although it appears in the text as two lines.
Note	Alerts the reader to exceptions or special conditions.
Important	Alerts the reader to critical information.
Warning	Used in situations where you can overwrite or corrupt data, unless you follow the instructions.



Chapter 1

Installation Overview

This chapter covers preparatory information for installing Q/LinQ.

Installation Introduction **8**

Minimum Requirements **9**

Installation Utility **10**

Installation Introduction

Use these instructions to install the Q/LinQ software on either UNIX or Windows servers. The Q/LinQ software functions in both Oracle and Progress database environments.

Audience

These instructions are for MFG/PRO system administrators who manage the MFG/PRO database and are familiar with the UNIX or Windows environments, networking, and Progress.

Installation Errata

In addition to these instructions, you may receive a supplementary errata sheet with changes and additional instructions. Check your product package.

Even if an errata sheet is included with your product package, QAD recommends that you download the most up-to-date errata sheet from the QAD Web site. New information may have been added to the errata sheet since your product was shipped.

To ensure a smooth implementation, QAD installation instructions are periodically updated. To determine whether your installation instructions have been updated, refer to the QAD Web site. Compare the item number listed on your guide with the number listed on the QAD Web site. If your guide has been updated, download and use the most recent version.

Q/LinQ CD-ROM

For details, see Chapter 7, “Initialization Files and Installation Directories,” on page 59.

The CD-ROM for Q/LinQ contains all of the Q/LinQ directories plus additional development directories.

Installations

Q/LinQ includes both Progress and non-Progress programs. The Progress programs are installed as part of the MFG/PRO installation.

This guide provides operating system-specific instructions for installing the non-Progress programs:

- Chapter 2, “Installation on UNIX,” on page 15
- Chapter 4, “Installation on Windows NT,” on page 33

In addition, use the appropriate chapter to install and set up mapping software and communication with external applications:

- Chapter 3, “Communication and Mapping Setup on UNIX,” on page 21
- Chapter 5, “Communication and Mapping Setup on Windows,” on page 41

Minimum Requirements

Table 1.1 lists the database server requirements specific to Q/LinQ. For MFG/PRO minimum requirements, refer to the appropriate installation guide for your system.

Table 1.1 Database Server Requirements

Requirement	Description
Progress	For MFG/PRO 9.0, eB, and eB2, the Progress version required by MFG/PRO is sufficient. At least one license of Progress 4GL or ProVISION.
MFG/PRO	Version 9.0, MFG/PRO eB, or MFG/PRO eB2 installed and configured.
Disk Space	The installation program verifies that at least 20 MB of space exists on the selected file system.

Table 1.2 lists the additional requirements for using MQSeries for message-oriented middleware communication.

Table 1.2 MQSeries Requirements

Requirement	Description
IBM MQSeries	Version 5.0 or above.

QAD does not provide a Java Virtual Machine (JVM) with Q/LinQ. JVMs can be downloaded, free of charge, from many third-party Web sites, including Sun Microsystems and Microsoft. Table 1.3 lists the client requirements for Java.

Table 1.3
Java Requirements

Requirement	Description
Java Plug-In	MFG/PRO 9.0 before Service Pack 6 and MFG/PRO eB before Service Pack 3 require Java Plug-In 1.1.8 or higher.
Java Runtime Environment (JRE), including the Java Plug-In	MFG/PRO 9.0 starting with Service Pack 6, MFG/PRO eB starting with Service Pack 3, and MFG/PRO eB2 require JRE 1.3 or higher.

Installation Utility

The installation for UNIX uses MFG/UTIL, a set of programs designed specifically for installing and managing MFG/PRO.

Keyboard Commands

When using MFG/UTIL in a character user interface, use the keyboard commands listed in Table 1.4.

Table 1.4
MFG/UTIL Keyboard Commands

Navigation Commands	Keyboard Entry	Control Key Entry	Description
Go	F1	Ctrl+X	Moves to next frame.
End	F4	Ctrl+E	Exits a frame, program, or menu.
Previous	F9 or Up Arrow	Ctrl+K	Retrieves previous record in a key data field and scrolls up in look-up browses.
Next	F10 or Down Arrow	Ctrl+J	Retrieves next record in a key data field and scrolls down in look-up browses.
Select	Space		Selects check boxes and on/off options.
Enter	Enter		Moves to next field within a frame.
Tab	Tab		Moves to next field within a frame.
Back Tab	Shift+Tab	Ctrl+U	Moves back one field within a frame.
Exit		Ctrl+C	Ends process.

Note In the character interface, button commands appear within angle brackets; for example: `< OK >`. To execute a button command, use the Tab key to move to the button and press Enter.

Log Files

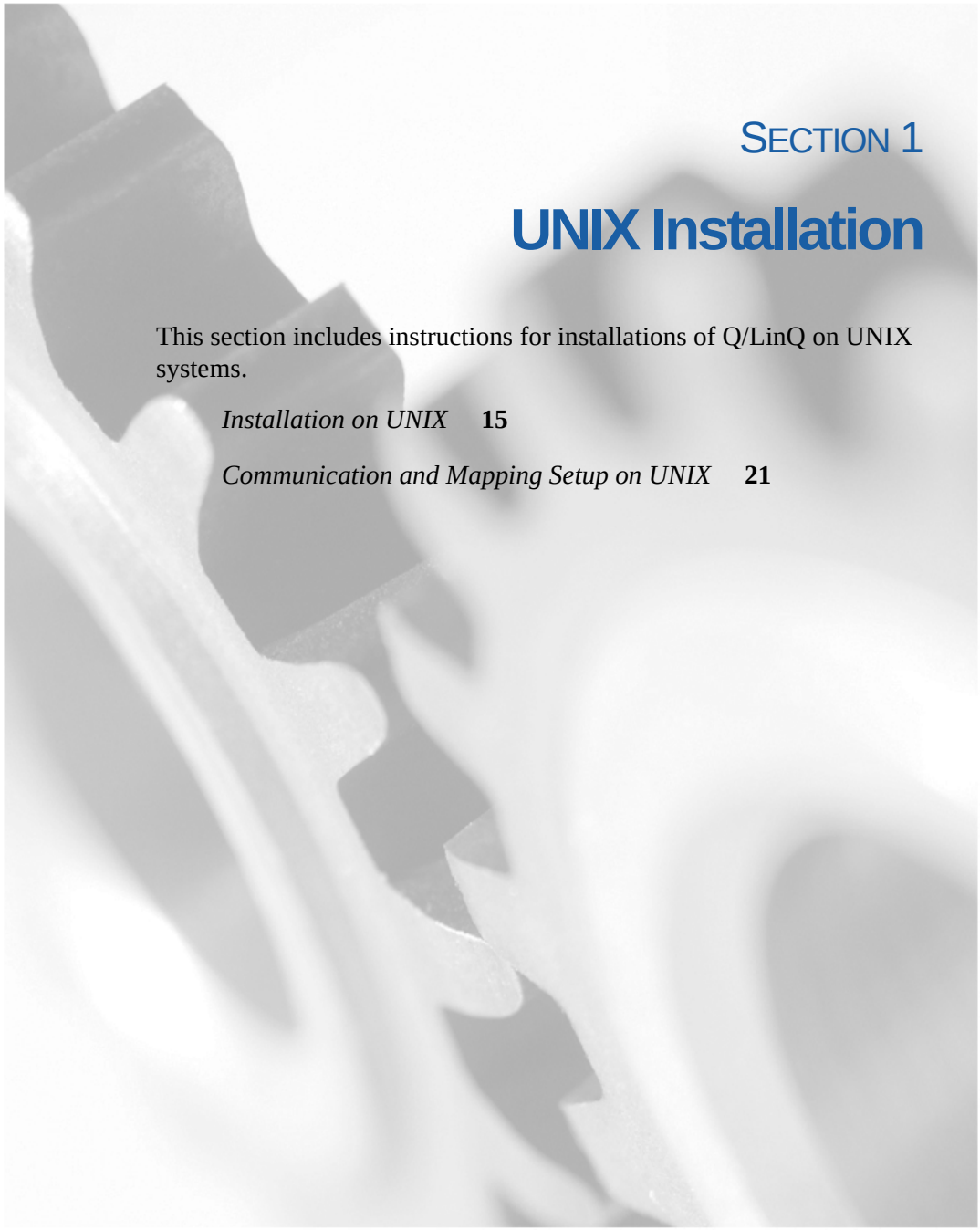
For UNIX systems, the installation creates the log files listed in Table 1.5.

Table 1.5 Installation Log Files

Utility	Log File Name	Directory Location
Installation Script	`qlinq.log`	Subdirectory `/log` under the installation directory
MFG/UTIL	`mfgutil.log`	MFG/PRO installation directory

Each time MFG/UTIL runs a significant, resource-intensive task such as compiling or loading a `.df` file, it creates a new log file. The most recent log file is always called `mfgutil.log`. Older log files are named with the convention `mfgutil.xxx`, where `xxx` is a number from 001 through 999. The lower the number, the older the file. For example, these files are listed in order from newest to oldest:

```
mfgutil.log
mfgutil.002
mfgutil.001
```



SECTION 1

UNIX Installation

This section includes instructions for installations of Q/LinQ on UNIX systems.

Installation on UNIX **15**

Communication and Mapping Setup on UNIX **21**

Chapter 2

Installation on UNIX

This chapter includes the instructions for UNIX installations for MFG/PRO 9.0, eB, and eB2. It covers the steps required to install Q/LinQ non-Progress programs in a UNIX environment where the Progress programs were installed with MFG/PRO.

Installation Overview **16**

Mounting the CD-ROM **16**

Running the Installation **17**

Storing the CD-ROM **19**

Modifying the PROPATH **19**

Registering Q/LinQ (MFG/PRO eB2 Only) **20**

Installation Overview

The following are key steps required for an installation of non-Progress Q/LinQ programs in a UNIX environment:

- Run the Q/LinQ UNIX installation.
- Modify the Q/LinQ client PROPATH to include the Q/LinQ directories.

See page 19.

To complete the installation and setup, see Chapter 3, “Communication and Mapping Setup on UNIX,” on page 21.

A separate service pack CD is not provided with Q/LinQ. If a service pack is available for your version of Q/LinQ, it is included on the standard product CD. Review the `readme.pdf` or `readme.htm` file included in the Q/LinQ installation directory for installation instructions specific to your CD and to learn about updates to the Q/LinQ release.

Mounting the CD-ROM

Follow these steps to mount the Q/LinQ installation CD-ROM:

1. Log on as the `root` user ID.
2. Create a CD-ROM directory if one does not already exist. These instructions refer to this directory as *`YourCDROMDir`*.
3. Place the Q/LinQ installation CD-ROM in the CD-ROM drive.
4. Mount the CD-ROM. The mount command differs from system to system. Table 2.1 lists commands for mounting the Q/LinQ CD-ROM.

Table 2.1 CD-ROM Mount Commands

Hardware	Mount Command
Sun	`volcheck cdrom`
HP	`mount /dev/dsk/`*`YourcdDevice`*`/cdrom`
DEC	`mount -t cdfs` *`YourcdDevice`*`/cdrom`
All others	Refer to your hardware system documentation or vendor for requirements for mounting a CD-ROM. You might be able to type `man mount` to determine the correct command.

Running the Installation

Follow these steps to install Q/LinQ on your server using the Q/LinQ installation script and MFG/UTIL:

1. Log on as user `mfg` under the group `qad`.

   Note You should have already created this user when you installed MFG/PRO.

2. Locate the Progress directory on the server and make note of its location.

3. From a UNIX prompt, change to the `unix` directory on the drive containing the Q/LinQ CD-ROM:

   `cd` *`YourCDROMDir`*`/unix`

4. Execute the script `install`:

   `./install`

5. At the following prompt, enter the path to the Progress installation directory; accept the default if it is correct:

   `Enter Progress installation directory.`

6. Tab to Next, press Enter.

7. Use standard terminal types when installing Q/LinQ. For example, for terminal type VT-100, set the TERM variable to vt100 during the installation:

   ```
   Enter Your terminal type,
      [default = CurrentSetting]:
   ```

8. At the following prompt, enter the installation directory for Q/LinQ:

   ```
   Enter the full path of the directory where you want to
   install Q/LinQ.
   ```

   The Q/LinQ programs are loaded into this directory.

   Note Ensure that the user `mfg` has permission to write to the Q/LinQ installation directory.

9. Tab to Next, press Enter.

10 At the following prompt, use the arrow keys to move to the platforms that will run Q/LinQ. Press Enter to select a platform:

```
Select the platforms to install Q/LinQ on. Select the
highest OS version which is less than or equal to the
current OS version.
```

For example, for a Solaris 2.6 system, select `solrs25`.

11 Tab to Finish, press Enter.

12 Wait while the Q/LinQ installation determines disk space availability. If there is not enough space, the installation stops. Obtain adequate disk space and restart the installation.

A list of the files being installed displays on the screen.

13 At the following prompt, enter Yes to delete the temporary files the Q/LinQ installation created:

```
Do you want to clean up the /tmp stage directory?
```

14 This completes the process of loading the Q/LinQ files onto the server. Use a text editor to view the installation log:

`QlinqInstallDir``/log/qlinq.log`

15 Ensure that the Q/LinQ directories listed in `qqapi.ini` are available to all users.

- `qqArchiveDir`
- `qqEditDir`
- `qqExpLog`
- `qqImpLog`
- `qqWorkDir`

a To avoid Q/LinQ runtime errors, enable write permissions for these directories for all users.

b To avoid file name clashes at runtime, ensure that the path to each of these directories is unique.

You can use a single `qqapi.ini` file for all users unless your computer environment requires that different users access the Q/LinQ directories using different paths. In that case, a separate `qqapi.ini` file is required for each path.

16 Review the `readme.pdf` or `readme.htm` file included in the Q/LinQ installation directory for installation instructions specific to your CD and to learn about updates to the Q/LinQ release.

Storing the CD-ROM

1 Log on as the `root` user ID.

2 Unmount the CD-ROM by typing the following command:

```
umount /YourCDROMDir
```

3 Remove the CD-ROM from your CD-ROM drive and return it to the case for safekeeping.

Modifying the PROPATH

The Q/LinQ installation creates the Q/LinQ initialization file, `qqapi.ini`. This file *must* be included in the MFG/PRO PROPATH in order to use Q/LinQ functions.

Warning If the `qqapi.ini` file is not included in the MFG/PRO PROPATH, some Q/LinQ functions that use one of the directories specified in `qqapi.ini` will fail with the warning message:

```
File qqapi.ini not found.
```

If you use different PROPATHs for different users, ensure that `qqapi.ini` is available to all Q/LinQ clients.

Use one of these methods to make the file available:

- Copy `qqapi.ini` from the *`QlinqInstallDir`* directory and place it in a directory already listed in your MFG/PRO PROPATH.

  OR

- Follow these steps to add the path to the *`QlinqInstallDir`* directory in your MFG/PRO PROPATH:

  a From the MFG/UTIL Configure menu, choose DLC and PROPATH Variables.

  b In the PROPATH field, specify the directory location of `qqapi.ini`.

c Insert this information at the end of the PROPATH and separate each location reference with a comma. The following example shows the PROPATH for a Q/LinQ initialization file in the `QlinqInstallDir`:

```
.,/existingPropath,/QlinqInstallDir,
```

For MFG/PRO 9.0 and eB, continue the installation by completing the instructions in Chapter 3, “Communication and Mapping Setup on UNIX,” on page 21. For MFG/PRO eB2, review the next section before continuing.

Registering Q/LinQ (MFG/PRO eB2 Only)

Beginning with MFG/PRO eB2, Q/LinQ has a separate license code that must be registered in MFG/PRO. The license code should be included with your product package.

If you are installing Q/LinQ as part of an MFG/PRO installation, all of the required license codes can be entered at the same time. This process is described in the MFG/PRO eB2 installation guide for your system.

If you purchased Q/LinQ separately and are installing it after MFG/PRO has been set up, you must register the license codes and assign users access before the new programs can be used. Use License Registration (36.16.10.1) to add the new license codes. You can assign user access as part of registering the license or in User Maintenance (36.3.18). See *User Guide Volume 9: Manager Functions* for details on using these programs.

Chapter 3

Communication and Mapping Setup on UNIX

This chapter is required for all UNIX installations. It covers the steps required to set up mapping software and communication with external applications in UNIX environments.

Overview **22**

Setting Up Non-MOM Communication **22**

Setting Up MOM Communication **23**

Setting Up Mapping **29**

Completing Setup **30**

Overview

This chapter covers installation and setup of communication and mapping software.

See page 22.

- If you are not using MQSeries message-oriented middleware (MOM), set up non-MOM communication with external application adapters.

See page 23.

- If you are using MQSeries MOM, install it and set up MOM adapter communication.

See page 29.

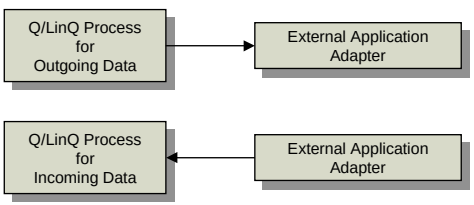
- If you are using the QAD-supplied mapping based on Extended Stylesheet Language Transformation (XSLT) technology, configure it for use.

Setting Up Non-MOM Communication

This section covers setting up communication with external applications when message-oriented middleware (MOM) is *not* being used. For MOM communication, see “Setting Up MOM Communication” on page 23.

Q/LinQ communicates with non-Progress external applications through application-specific adapters. The adapter is the communication bridge between Q/LinQ and the external application.

Fig. 3.1
Non-MOM Communication



Setting up non-MOM communication with the adapter for an external application includes these steps:

- Allocate network ports.
- Specify the ports to Q/LinQ.

Allocate network ports in the `services` file on the Q/LinQ host computer, usually located in the `/etc` or `/etc/inet` directory. Reserve ports for both Q/LinQ and any adapters not using MOM.

1. Allocate one network port number for each direction of concurrent document exchange with each registered external application. That is, allocate one port for an application unless exporting and importing documents concurrently.
2. Set up Q/LinQ Control by updating required fields. The control records must exist before you can register an application.

See page 104.

3. Specify the Q/LinQ-adapter ports in the Messaging API Socket Parameters frame of Register External Application (36.8.1.1; 36.8.1 in 9.0), so that import and export sessions reference the adapter host and the appropriate Q/LinQ-adapter network port numbers.

See “Registering External Applications” on page 107 for details.

Note You should review the entire chapter on setting up Q/LinQ before proceeding with these steps.

See Chapter 11.

Setting Up MOM Communication

Message-oriented middleware (MOM) is a server program that mediates communication between multiple client applications. Q/LinQ supports MQSeries MOM from IBM for message-oriented communication between Q/LinQ and non-MFG/PRO applications.

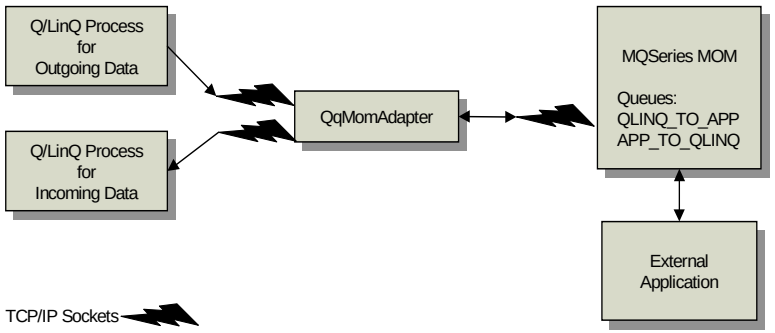
Q/LinQ also includes an adapter (`QqMomAdapter`) for communication between Q/LinQ and MQSeries.

During an installation, `QqMomAdapter` and its associated `.ini`, `.jar`, and `.class` files are placed on the Q/LinQ host computer in the `mom` directory. The adapter can run from the Q/LinQ machine, the MOM machine, or a third machine.

For details on the `.ini` file, see “Q/LinQ Initialization Files” on page 60.

`QqMomAdapter` is a TCP/IP listener that handles multiple client connections, which are the Q/LinQ import and export sessions that the `QqMomAdapter` process is dedicated to. When Q/LinQ completes a session, it disconnects from `QqMomAdapter`, which then goes back into listening mode waiting for the Q/LinQ session to connect again. `QqMomAdapter` runs continually, regardless of whether any Q/LinQ sessions are connected to it.

Fig. 3.2
MQSeries MOM Communication



Setting up MOM communication with a non-MFG/PRO application includes these steps:

- Install and configure the MOM software.
- Allocate network ports.
- Configure Q/LinQ clients for the adapter.
- Configure the non-MFG/PRO application.

Important The following instructions refer to the Q/LinQ client machine that is running the Q/LinQ MOM adapter as the MOM client.

Install and Configure MQSeries MOM

MQSeries MOM requires a queue manager, channels, and queues. Create queues for each application that will communicate with Q/LinQ using MQSeries and `QqMomAdapter`. Each queue is a point-to-point logical connection that provides one-way communication between `QqMomAdapter` and MQSeries.

- If Q/LinQ imports data from an application, set up a Q/LinQ receiving queue for that application.
- If Q/LinQ exports data to an application, set up a Q/LinQ sending queue for that application.
- When Q/LinQ both exports data to and imports data from an application, set up two queues for that application: one for sending and the other for receiving.

Important Once a message is retrieved, it is removed from that queue. If two or more clients are set up to retrieve messages from the same queue, they will be in competition, and neither client will receive all the messages in the queue. Configure clients so that only one can receive messages from each queue. The same consideration does not apply to placing messages into a queue; multiple clients can place messages into the same queue without conflict.

Use the following instructions to install MQSeries, create an MQSeries queue manager, create MQSeries channels, and create MQSeries queues:

1. Using the MQSeries documentation, install the MQSeries server.

   Note On most UNIX platforms, the computer hosting the server must be reconfigured through changes to the `/etc/hosts`, `/etc/services`, and `/etc/inetd.conf` files in order to start an MQSeries channel listener automatically. In other systems, this is done explicitly through a command invoked from the MQSeries command interpreter. Consult the MQSeries platform-specific installation instructions for details.

2. Using the MQSeries documentation, install the MQ Java Client on the Q/LinQ client machine that will run `QqMomAdapter`.

3. Install a Java Virtual Machine (JVM) on the Q/LinQ client machine that will run `QqMomAdapter`.

   QAD does not provide a JVM with Q/LinQ. JVMs can be downloaded, free of charge, from many third-party Web sites, including Sun Microsystems and Microsoft.

   See “Minimum Requirements” on page 9 for the version of Java required.

4. Create and start an MQSeries queue manager called QADQM:

   ```
   crtmqm -q -u SYSTEM.DEAD.LETTER.QUEUE. QADQM
   strmqm QADQM
   ```

   Note You can use a different name for the queue manager, but you must then change the `QManager` parameter in `QqMomAdapter.ini` to reference the new name.

5. Start an MQSeries command interpreter session. In this example, QADQM is the queue manager created in step 4:

   ```
   runmqsc QADQM
   ```

6 In the MQSeries command interpreter, create an MQSeries channel and name it QLINQ.CHANNEL. This channel must allow two-way communication between the MOM client and the MQServer. Enter the following command as one line:

```
define channel ('QLINQ.CHANNEL')chltype(svrconn)
   trptype(tcp) replace
```

Note You can use a different name for the MQSeries channel, but you must then change the `channel` parameter in `QqMomAdapter.ini` to reference the new name.

7 In the MQSeries command interpreter, create application queues. To prevent confusion with queue names, use a convention that denotes the applications using the queue and the direction of communications. MQSeries naming conventions use all capital letters. These examples create the queues QLINQ_TO_APP and APP_TO_QLINQ:

```
define qlocal (QLINQ_TO_APP) replace
define qlocal (APP_TO_QLINQ) replace
```

8 End the MQSeries command interpreter session that was started in step 6:

```
end
```

See Note following Step 1 on page 25.

9 If necessary for your platform, start an MQSeries listener process to service `QqMomAdapter` requests:

```
runmqlsr -t tcp -m QADQM -p 1414
```

`1414` is the port, defined in `QqMomAdapter.ini`, through which MQSeries and `QqMomAdapter` communicate.

Allocate Network Ports on the Q/LinQ Host Computer

Allocate network ports in the `services` file on the Q/LinQ host computer, usually located in the `/etc` or `/etc/inet` directory. Reserve ports for both Q/LinQ and `QqMomAdapter`.

See Figure 3.2 on page 24.

1 Allocate one network port number for each import or export queue in MQSeries with which Q/LinQ will communicate. For example, allocate one for the export queue QLINQ_TO_APP and one for the import queue APP_TO_QLINQ.

2 Allocate one network port number for communications between `QqMomAdapter` and MQSeries. The MQSeries default port is 1414.

3 Specify the Q/LinQ `QqMomAdapter` ports in Register External Application (36.8.1.1; 36.8.1 in 9.0), Messaging API Socket Parameters, so that import and export sessions reference the `QqMomAdapter` host and the appropriate Q/LinQ-adapter network port number.

4 Specify the `QqMomAdapter` MQSeries port. See step 4 in the following section.

Configure the MOM Client

Use the following instructions to configure a Q/LinQ client machine (MOM client) to run `QqMomAdapter`:

1 Copy the contents of the `mom` directory from the Q/LinQ host computer to the MOM client.

2 Set the system environment variable `PATH` to include the `bin` directory for the JVM. This puts the Java commands into the command search path.

3 Include the paths to the following elements in the system environment variable `CLASSPATH`:

- The MQSeries client libraries, specifically, the designated `.jar` files within the `com.ibm` package.
- The `mom/` or `resource/mom/` directory on the Q/LinQ host computer. This directory contains `QqMomAdapter` and its associated files.
- The `classes/` or `resource/classes/` directory on the Q/LinQ host computer. This directory contains `QqSend` and `QqRecv` with associated files.
- These `.jar` files in the `mom/` or `resource/mom/` directory:

  `momapi.jar`

  `qqmom.jar`

- These `.jar` files in the `classes/` or `resource/classes/` directory:

  `qqapi.jar`

  `qadgen.jar`

  `qqutil.jar`

  `qqtest.jar`

See “Setting Up Mapping” on page 29.

Note If you are using the XSLT mapper as well as the MOM adapter, these `CLASSPATH` entries should be added to those required for XSLT.

4 In the `mom` directory, use a text editor to modify the `[MQSeries]` section of `QqMomAdapter.ini`.

See “Allocate Network Ports on the Q/LinQ Host Computer” on page 26, step 4.

Enter system-specific information for the host and port:

`host:` Enter the IP address or machine name of the MQSeries server machine.

`port:` Enter the port number defined for communication between `QqMomAdapter` and MQSeries.

5 In the MOM client `services` file, usually located in the `/etc/inet` or `/etc` directory, reserve ports for both Q/LinQ and `QqMomAdapter`.

Configure the Non-MFG/PRO Application

Configure the MOM adapter or the communication layer of the non-MFG/PRO application to use the appropriate MQSeries queues for importing and exporting.

Starting and Stopping the MOM Adapter

For details about starting and stopping the MOM adapter, see the section in “Starting and Stopping the MOM Adapter” on page 119.

Setting Up Mapping

Q/LinQ provides its own XML data-mapping capability built using Extended Stylesheet Language Transformation (XSLT), the industry standard for XML-based data transformation. XSLT was defined by the World Wide Web Consortium (W3C) as a language for mapping XML data. Therefore, it is a natural choice for manipulating and formatting MFG/PRO data packaged as business object documents (BODs) according to the Open Application Group's (OAG) published specifications.

XSLT is also implemented and supported by open-source initiatives such as the Apache Software Project. By taking advantage of available open-source technology, QAD has eliminated the need for MFG/PRO customers to license any third-party mapping products in order to use OAG BODs for systems integration purposes.

Follow these steps to enable Q/LinQ to use the XSLT mapping engine:

1. Include the paths to the following elements in the system environment variable `CLASSPATH`:
   - The JVM core Java classes. These are typically in `lib/classes.zip` under the Java install directory.
   - The `classes/` or `resource/classes/` directory in the Q/LinQ install directory.
   - The following `.jar` files in the Q/LinQ `classes/` or `resource/classes/` directory:

     ```
     mapper.jar
     xalan.jar
     xerces.jar
     ```

   Note If you are using the MOM adapter as well as XSLT mapper, the `CLASSPATH` entries in step 1 should be added to those required for the MOM adapter.

   See step 3 on page 27.

2. See *External Interface Guide: Logistics* for instructions on setting up Q/LinQ to support the various QAD-provided BODs required for integration with an external Logistics application.

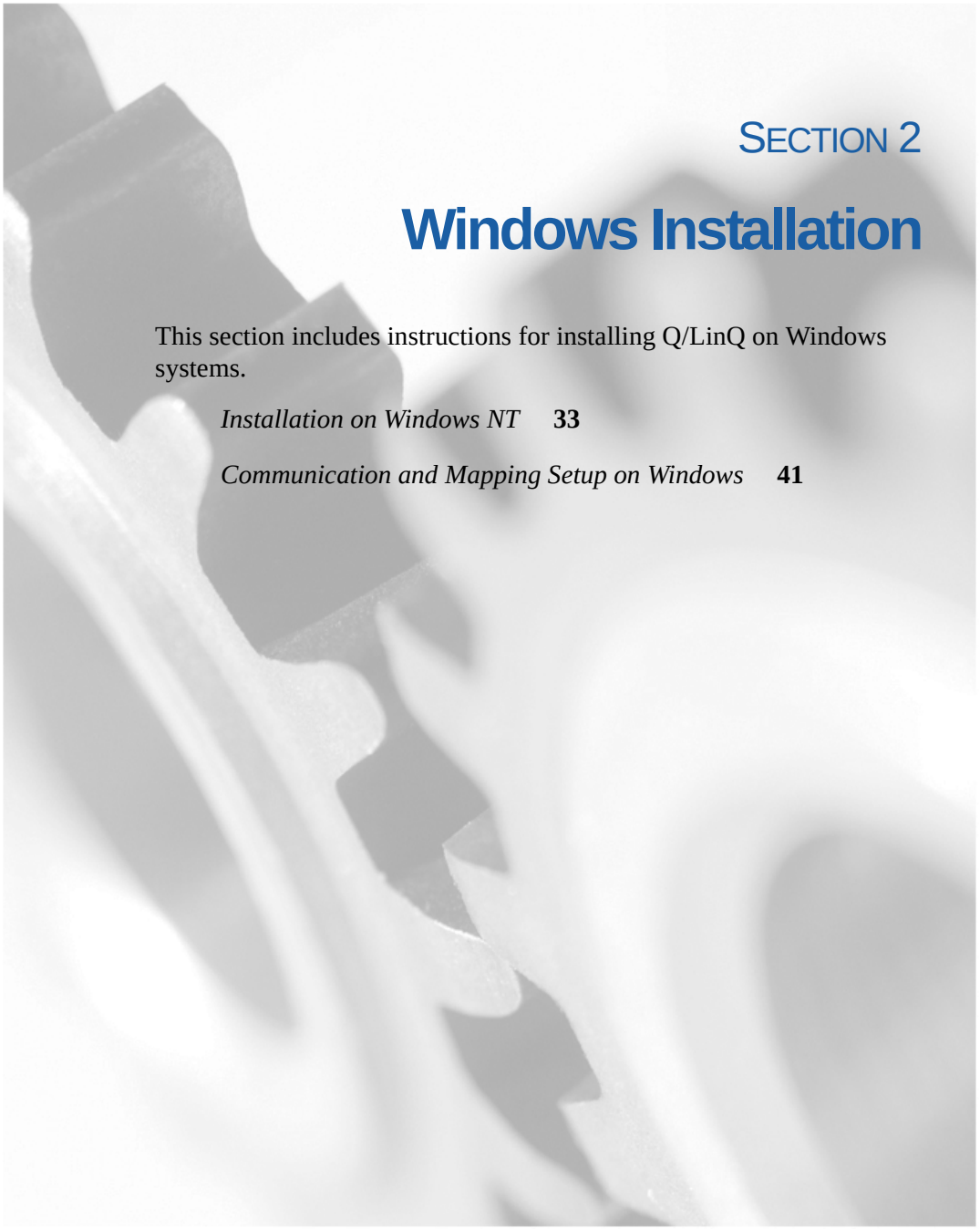
3 In order to obtain detailed trace output from the XSLT mapper for debugging purposes, add the following mapping parameter and value to any Q/LinQ import or export specification set up to use XSLT:

- Mapping Parameter Name: `DEBUG`
- Mapping Parameter Value: <debug file name, for example `debug.txt`>

A debug file containing detailed output and diagnostic messages will be created in the current directory whenever Q/LinQ calls the XSLT mapper to map the designated document type.

Completing Setup

When you have completed the tasks in this chapter, you should continue with Chapter 11, “Setting Up Q/LinQ,” and then use the details in the section on “Testing Communication” on page 122 to ensure that your set up works correctly.



SECTION 2

Windows Installation

This section includes instructions for installing Q/LinQ on Windows systems.

Installation on Windows NT **33**

Communication and Mapping Setup on Windows **41**

Chapter 4

Installation on Windows NT

This chapter includes the instructions for Windows installations for MFG/PRO 9.0, eB, and eB2. It covers the steps required to install Q/LinQ non-Progress programs in a Windows NT environment where the Progress programs were installed with MFG/PRO.

Installation Overview **34**

Running the File Server Installation **34**

Running the Client Installation **37**

Modifying the Client PROPATH (Optional) **39**

Registering Q/LinQ (MFG/PRO eB2 Only) **40**

Installation Overview

The following are key steps required for an installation of non-Progress Q/LinQ programs in a Windows environment.

- Run the Q/LinQ Windows server installation.
- From the Q/LinQ Windows server installation directory, run the Q/LinQ Windows client installation for each client.

See page 37.

- Modify the client PROPATH.

See page 39.

To complete the installation and setup, see Chapter 5, “Communication and Mapping Setup on Windows,” on page 41.

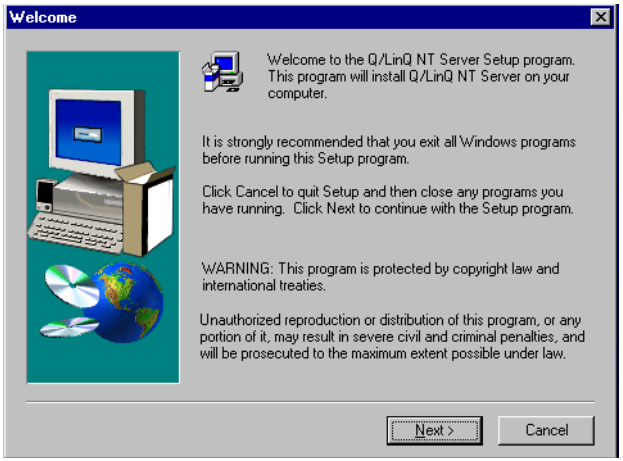
A separate service pack CD is not provided with Q/LinQ. If a service pack is available for your version of Q/LinQ, it is included on the standard product CD. Review the `readme.pdf` or `readme.htm` file included in the Q/LinQ installation directory for installation instructions specific to your CD and to learn about updates to the Q/LinQ release.

Running the File Server Installation

1. Insert the Q/LinQ CD-ROM in the CD-ROM drive.
2. Run the `setup.exe` program using File Manager, Windows Explorer, or the Run command in the Windows Start menu.

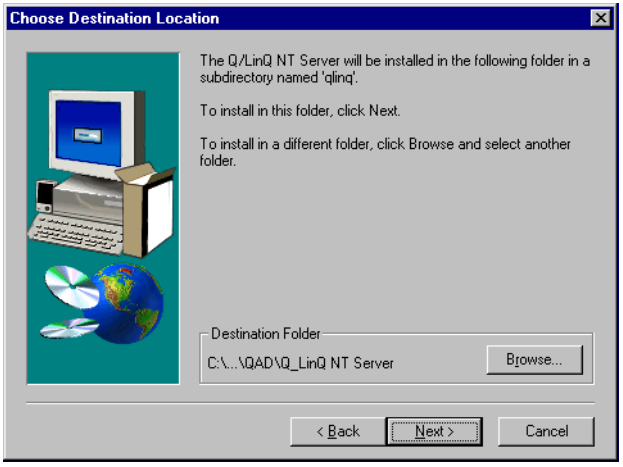
   `CD-ROMDriveName``:\nt\setup.exe`

3. At the Welcome window, click Next.

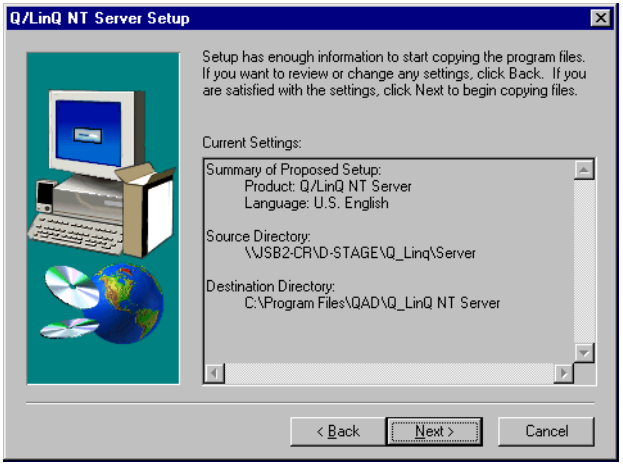


4 Specify the directory in which you want to install the Q/LinQ server. Accept the default, or use the Browse button to select an alternate directory. Click Next to continue.

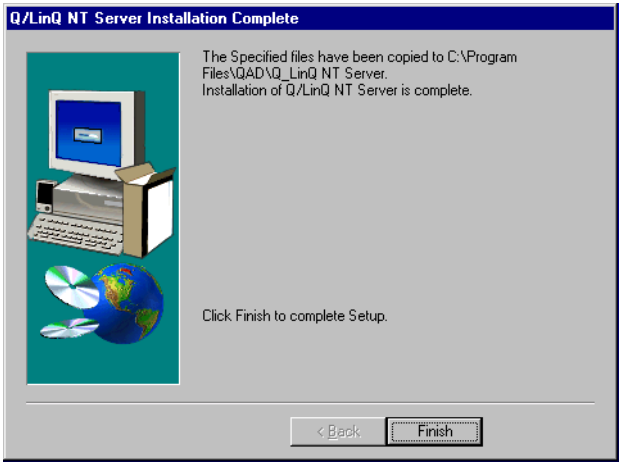
Important Make sure that the machines you plan to set up as Q/LinQ clients can access this directory. The Q/LinQ client setup executable is run from the Q/LinQ server installation directory.



5 Verify that the installation information is correct. If any of the information is incorrect, click Back. Otherwise, click Next.



6 When the Q/LinQ NT Server Installation Complete window displays, click Finish to complete the server installation.



7 Ensure that the Q/LinQ directories listed in `qqapi.ini` are available to all users.

- `qqArchiveDir`
- `qqEditDir`

- `qqExpLog`
- `qqImpLog`
- `qqWorkDir`

a To avoid Q/LinQ runtime errors, enable write permissions for these directories for all users.

b To avoid file name clashes at runtime, ensure that the path to each of these directories is unique.

A single `qqapi.ini` file can be used for all users unless your computer environment requires that different users must access Q/LinQ directories using different paths. In that case, a separate `qqapi.ini` file is required for each path.

Running the Client Installation

1 Access the Q/LinQ server installation directory from the Windows client machine and run `setup.exe`.

In the following example, *`QlinqInstallDir`* refers to the Q/LinQ installation directory on the Q/LinQ server. *`MappedDrive`* refers to the Windows client drive used to map to the Q/LinQ server installation directory.

See step 4 on page 35.

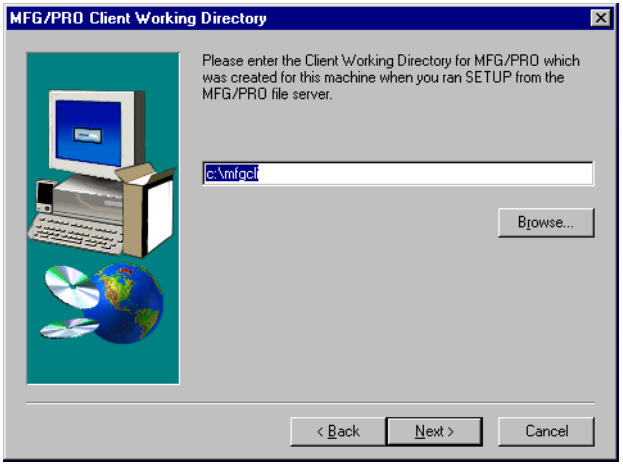
`MappedDrive``:\`*`QlinqInstallDir`*`\setup.exe`

2 At the Welcome window, click Next.

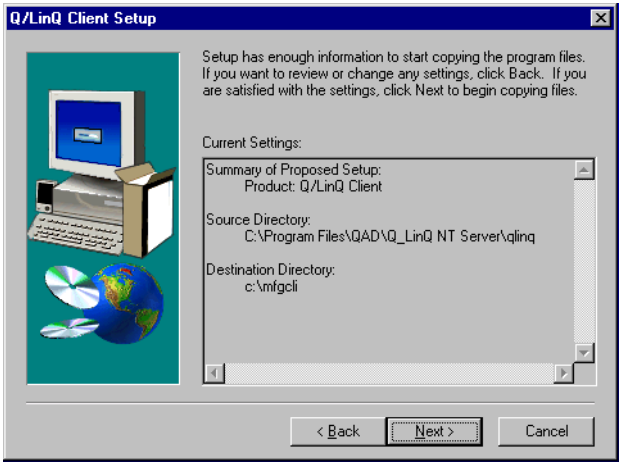
- See the Windows GUI and Character Client Setup chapters in the MFG/PRO installation guide for your system.

3 Specify the path to the MFG/PRO working directory on the client. Accept the default if it is correct; otherwise, use the Browse button to select a different directory. Click Next when you are ready to continue.

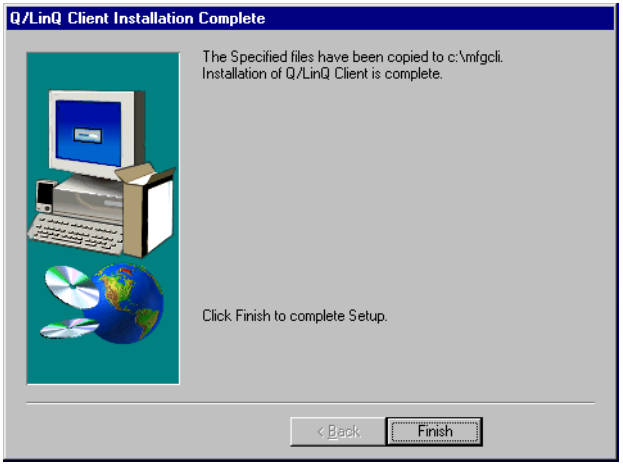
Note This directory was created during the MFG/PRO Windows client setup process.



4 Verify that the installation information is correct. If any information is incorrect, click Back. Otherwise, click Next.



5 When the Q/LinQ Client Installation Complete window displays, click Finish to complete the client installation.



6 Review the `readme.pdf` or `readme.htm` file included in the *`QlinqInstallDir`* location to obtain installation instructions specific to your CD and learn about the contents of the Q/LinQ release.

Modifying the Client PROPATH (Optional)

During the Q/LinQ client installation, the path to `qqapi.ini` is automatically included in the PROPATH of the local MFG/PRO startup icons.

Follow these steps to modify or adjust the PROPATH for a Q/LinQ client, if required.

1 Right-click on the local MFG/PRO startup icon and select Properties.

2 In the Properties Window, choose the Shortcut tab. In the Target field, locate the `-ininame` variable and record the path to the `-ininame` file.

Usually, this file is located in your MFG/PRO client working directory and is called `progress.svg`, `progress.vga`, or `progress.chr`, depending on the display type.

3 Open `-ininame` in a text editor.

4 In the PROPATH section, enter the path to `qqapi.ini` after the MFG/PRO directory paths.

```
PROPATH=
.,C:\mfgcli\images.pl,C:\mfgguifs,C:\mfgguifs\us\bbi,
    C:\mfgcli
```

5 Save your changes and exit the text editor.

For MFG/PRO 9.0 and eB, continue the installation by completing the instructions in Chapter 5, “Communication and Mapping Setup on Windows,” on page 41. For MFG/PRO eB2, review the next section before continuing.

Registering Q/LinQ (MFG/PRO eB2 Only)

Beginning with MFG/PRO eB2, Q/LinQ has a separate license code that must be registered in MFG/PRO. The license code should be included with your product package.

If you are installing Q/LinQ as part of an MFG/PRO installation, all of the required license codes can be entered at the same time. This process is described in the MFG/PRO eB2 installation guide for your system.

If you purchased Q/LinQ separately and are installing it after MFG/PRO has been set up, you must register the license codes and assign users access before the new programs can be used. Use License Registration (36.16.10.1) to add the new license codes. You can assign user access as part of registering the license or in User Maintenance (36.3.18). See *User Guide Volume 9: Manager Functions* for details on using these programs.

Chapter 5

Communication and Mapping Setup on Windows

This chapter is required for all Windows installations. It covers the steps required to set up mapping software and communication with external applications in Windows environments.

Overview **42**

Setting Up Non-MOM Communication **42**

Setting Up MOM Communication **43**

Setting Up Mapping **50**

Completing Setup **51**

Overview

This chapter covers installation and setup of communication and mapping software.

See page 42.

- If you are not using MQSeries message-oriented middleware (MOM), set up non-MOM communication with external application adapters.

See page 43.

- If you are using MQSeries MOM, install it and set up MOM adapter communication.

See page 50.

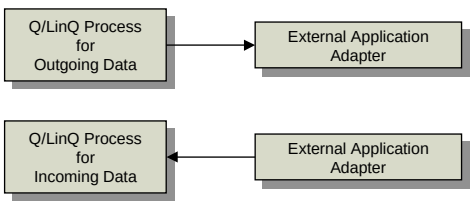
- If you are using the QAD-supplied mapping based on Extended Stylesheet Language Transformation (XSLT) technology, configure it for use.

Setting Up Non-MOM Communication

This section covers setting up communication with external applications when message-oriented middleware (MOM) is *not* being used. For MOM communication, see “Setting Up MOM Communication” on page 43.

Q/LinQ communicates with non-Progress external applications through application-specific adapters. The adapter is the communication bridge between Q/LinQ and the external application.

Fig. 5.1
Non-MOM Communication



Setting up non-MOM communication with the adapter for an external application includes these steps:

- Allocate network ports.
- Specify the ports to Q/LinQ.

Allocate network ports in the `services` file which is usually located in the `c:\winnt\system32\drivers\etc` directory on the Q/LinQ host computer. Reserve ports for both Q/LinQ and any adapters not using MOM.

1. Set up Q/LinQ Control by updating required fields. The control records must exist before you can register an application.

See page 104.

2. Specify the Q/LinQ-adapter ports in the Messaging API Socket Parameters frame of Register External Application (36.8.1.1; 36.8.1 in 9.0), so that import and export sessions reference the adapter host and the appropriate Q/LinQ-adapter network port numbers.

See “Registering External Applications” on page 107 for details.

Note You should review the entire chapter on setting up Q/LinQ before proceeding with these steps.

See Chapter 11.

Setting Up MOM Communication

Message-oriented middleware (MOM) is a server program that mediates communication between multiple client applications. Q/LinQ supports MQSeries MOM from IBM for message-oriented communication between Q/LinQ and non-MFG/PRO applications.

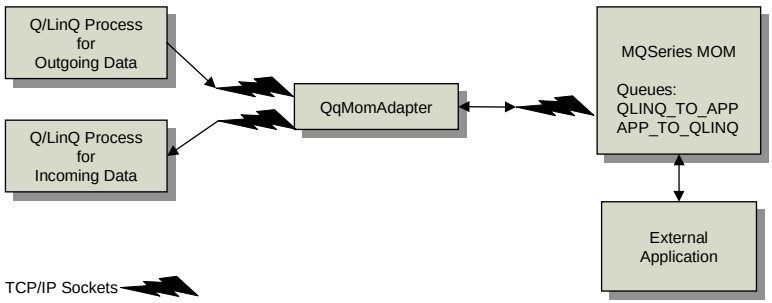
Q/LinQ also includes an adapter (`QqMomAdapter`) for communication between Q/LinQ and MQSeries.

During installation, `QqMomAdapter` and its associated `.ini`, `.jar`, and `.class` files are placed on the Q/LinQ server in the `mom` directory. The adapter can run from the Q/LinQ machine, the MOM machine, or a third machine.

For details on the `.ini` file, see “Q/LinQ Initialization Files” on page 60.

`QqMomAdapter` is a TCP/IP listener that handles multiple client connections, which are the Q/LinQ import and export sessions that the `QqMomAdapter` process is dedicated to. When Q/LinQ completes a session, it disconnects from `QqMomAdapter`, which then goes back into listening mode waiting for the Q/LinQ session to connect again. `QqMomAdapter` runs continually, regardless of whether any Q/LinQ sessions are connected to it.

Fig. 5.2
MQSeries MOM Communication



Setting up MOM communication with a non-MFG/PRO application includes these steps:

- Install and configure the MOM software.
- Allocate network ports.
- Configure Q/LinQ clients for the adapter.
- Configure the non-MFG/PRO application.

Important The following instructions refer to the Q/LinQ client running the Q/LinQ MOM adapter as the MOM client.

Install and Configure MQSeries MOM

MQSeries MOM requires a queue manager, channels, and queues. Create queues for each application that will communicate with Q/LinQ using MQSeries and `QqMomAdapter`. Each queue is a point-to-point logical connection that provides one-way communication between `QqMomAdapter` and MQSeries.

- If Q/LinQ imports data from an application, set up a Q/LinQ receiving queue for that application.
- If Q/LinQ exports data to an application, set up a Q/LinQ sending queue for that application.
- When Q/LinQ both exports data to and imports data from an application, set up two queues for that application: one for sending and the other for receiving.

Important Once a message is retrieved from a queue, it is removed from that queue. If two or more clients are set up to retrieve messages from the same queue, they will be in competition, and neither client will receive all the messages in the queue. Configure clients so that only one can receive messages from each queue. The same consideration does not apply to placing messages into a queue; multiple clients can place messages into the same queue without conflict.

Use the following instructions to install MQSeries, create an MQSeries queue manager, create MQSeries channels, and create MQSeries queues:

1. Using the MQSeries documentation, install the MQSeries server.
2. Using the MQSeries documentation, install the MQ Java Client on the Q/LinQ client machine that will run `QqMomAdapter`.
3. Install a Java Virtual Machine (JVM) on the Q/LinQ client machine that will run `QqMomAdapter`.

   QAD does not provide a JVM with Q/LinQ. JVMs can be downloaded, free of charge, from many third-party Web sites, including Sun Microsystems and Microsoft.

   See “Minimum Requirements” on page 9 for the version of Java required.
4. From the DOS prompt, create and start an MQSeries queue manager called QADQM:

   ```
   crtmqm -q -u SYSTEM.DEAD.LETTER.QUEUE. QADQM
   strmqm QADQM
   ```

   Note You can use a different name for the queue manager, but you must then change the `QManager` parameter in `QqMomAdapter.ini` to reference the new name.
5. Start an MQSeries command interpreter session. In this example, QADQM is the queue manager created in step 4.

   ```
   runmqsc QADQM
   ```
6. In the MQSeries command interpreter, create an MQSeries channel and name it QLINQ.CHANNEL. This channel must allow two-way communication between the MOM client and the MQServer. Enter the following command as one line:

   ```
   define channel (‘QLINQ.CHANNEL’)
      chltype(svrconn) trptype(tcp) replace
   ```

Note You can use a different name for the MQSeries channel, but you must then change the `channel` parameter in `QqMomAdapter.ini` to reference the new name.

7 In the MQSeries command interpreter, create application queues. To prevent confusion with queue names, use a convention that denotes the applications using the queue and the direction of communications. MQSeries naming conventions use all capital letters. These examples create the queues QLINQ_TO_APP and APP_TO_QLINQ:

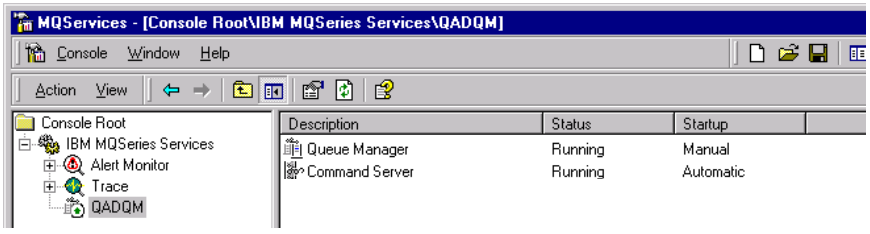
```
define qlocal (QLINQ_TO_APP) replace
define qlocal (APP_TO_QLINQ) replace
```

8 End the MQSeries command interpreter session that was started in step 5:

```
end
```

9 Create an MQSeries listener process. Click the Windows Start button and choose Programs. From the IBM MQSeries program folder, select MQSeries Services to display the MQServices Console.

Fig. 5.3
MQSeries Services Console



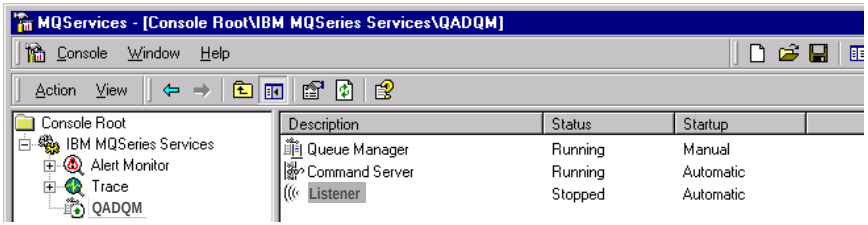
a Expand IBM MQSeries Services.

b Select QADQM.

c From the Action menu, choose New|Listener.

d Click OK in the dialog box to accept the defaults for the listener.

10 Start the MQSeries listener process to service `QqMomAdapter` requests. With Listener selected, from the Action menu, choose All Tasks|Start.

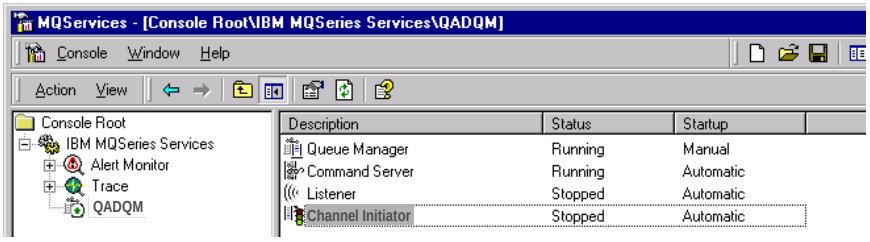


11 Create and start a channel initiator:

a Select QADQM.

b From the Action menu, choose New|Channel Initiator.

c Click OK in the dialog box to accept the defaults for the channel initiator.



d Select Channel Initiator.

e From the Action menu, choose All Tasks|Start.

12 Exit from the MQSeries Services Console.

Allocate Network Ports on the Q/LinQ Host Computer

Allocate network ports in the `services` file which is located in the `c:\winnt\system32\drivers\etc` directory of the Q/LinQ host computer. Reserve ports for both Q/LinQ and `QqMomAdapter`.

1 Allocate one network port number for each MQSeries import or export queue with which Q/LinQ will communicate; for example, allocate one for the export queue QLINQ_TO_APP and one for the import queue APP_TO_Q/LINQ.

See Figure 5.2 on page 44.

2 Allocate one network port number for communications between `QqMomAdapter` and MQSeries. The MQSeries default port is 1414.

3 Specify the Q/LinQ `QqMomAdapter` ports in the Messaging API Socket Parameters frame in Register External Application (36.8.1.1; 36.8.1 in 9.0), so that import and export sessions reference the `QqMomAdapter` host and the appropriate Q/LinQ-adapter network port number.

4 Specify the `QqMomAdapter` MQSeries ports. See step 4 in the following section.

Configure the MOM Client

Use the following instructions to configure a Q/LinQ client machine (MOM client) to run `QqMomAdapter`:

1 Copy the contents of the `mom` directory from the Q/LinQ server to the MOM client.

2 Set the system environment variable `PATH` to include the `bin` directory for the JVM. This puts the Java commands into the command search path.

3 Include the paths to the following elements in the system environment variable `CLASSPATH`:

- The MQSeries client libraries, specifically, the designated `.jar` files within the `com.ibm` package.
- The `mom\` or `resource\mom\` directory on the Q/LinQ server. This directory contains `QqMomAdapter` and its associated files.
- The `classes\` or `resource\classes\` directory on the Q/LinQ server. This directory contains `QqSend` and `QqRecv` with associated files.
- These `.jar` files in the `mom\` or `resource\mom\` directory:

  `momapi.jar`

  `qqmom.jar`

- These `.jar` files in the `classes\` or `resource\classes\` directory:

  `qqapi.jar`

  `qadgen.jar`

  `qqutil.jar`

  `qqtest.jar`

Note If you are using the XSLT mapper as well as the MOM adapter, these `CLASSPATH` entries should be added to those required for XSLT.

See “Setting Up Mapping” on page 50.

4 In the `mom` directory, use a text editor to modify the `[MQSeries]` section of `QqMomAdapter.ini`:

Enter system-specific information for host and port:

See “Allocate Network Ports on the Q/LinQ Host Computer” on page 47, step 4.

`host`: Enter the IP address or machine name of the MQSeries server machine.

`port`: Enter the port number defined for communication between MQSeries and `QqMomAdapter`.

5 In the MOM client `services` file, located in the `c:\winnt\system32\drivers\etc` directory, reserve ports for both Q/LinQ and `QqMomAdapter`.

Starting and Stopping the MOM Adapter

For details about starting and stopping the MOM adapter, see the section in “Starting and Stopping the MOM Adapter” on page 119.

Configure the Non-MFG/PRO Application

Configure the MOM adapter or the communication layer of the non-MFG/PRO application to use the appropriate MQSeries queues for importing and exporting.

Setting Up Mapping

Q/LinQ provides its own XML data-mapping capability built using Extended Stylesheet Language Transformation (XSLT), the industry standard for XML-based data transformation. XSLT was defined by the World Wide Web Consortium (W3C) as a language for mapping XML data. Therefore, it is a natural choice for manipulating and formatting MFG/PRO data packaged as business object documents (BODs) according to the Open Application Group's (OAG) published specifications.

XSLT is also implemented and supported by open-source initiatives such as the Apache project. By taking advantage of available open-source technology, QAD has eliminated the need for MFG/PRO customers to license any third-party mapping products in order to use OAG BODs for systems integration purposes.

Follow these steps to enable Q/LinQ to use the XSLT mapping engine:

1. Include the paths to the following elements in the system environment variable `CLASSPATH`:
   - The JVM core Java classes. These are typically in `lib/classes.zip` under the Java install directory.
   - The `classes/` or `resource/classes/` directory in the Q/LinQ install directory.
   - The following `.jar` files in the Q/LinQ `classes/` or `resource/classes/` directory:

     ```
     mapper.jar
     xalan.jar
     xerces.jar
     ```

See step 3 on page 48.

   Note If you are using the MOM adapter as well as XSLT mapper, the `CLASSPATH` entries in step 1 should be added to those required for the MOM adapter.

2. See *External Interface Guide: Logistics* for instructions on setting up Q/LinQ to support the various QAD-provided BODs required for integration with an external Logistics application.

3 In order to obtain detailed trace output from the XSLT mapper for debugging purposes, add the following mapping parameter and value to any Q/LinQ import or export specification set up to use XSLT:

- Mapping Parameter Name: `DEBUG`
- Mapping Parameter Value: <debug file name, for example `debug.txt`>

A debug file containing detailed output and diagnostic messages will be created in the current directory whenever Q/LinQ calls the XSLT mapper to map the designated document type.

Completing Setup

When you have completed the tasks in this chapter, you should continue with Chapter 11, “Setting Up Q/LinQ,” and then use the details in the section on “Testing Communication” on page 122 to ensure that your set up works correctly.

SECTION 3

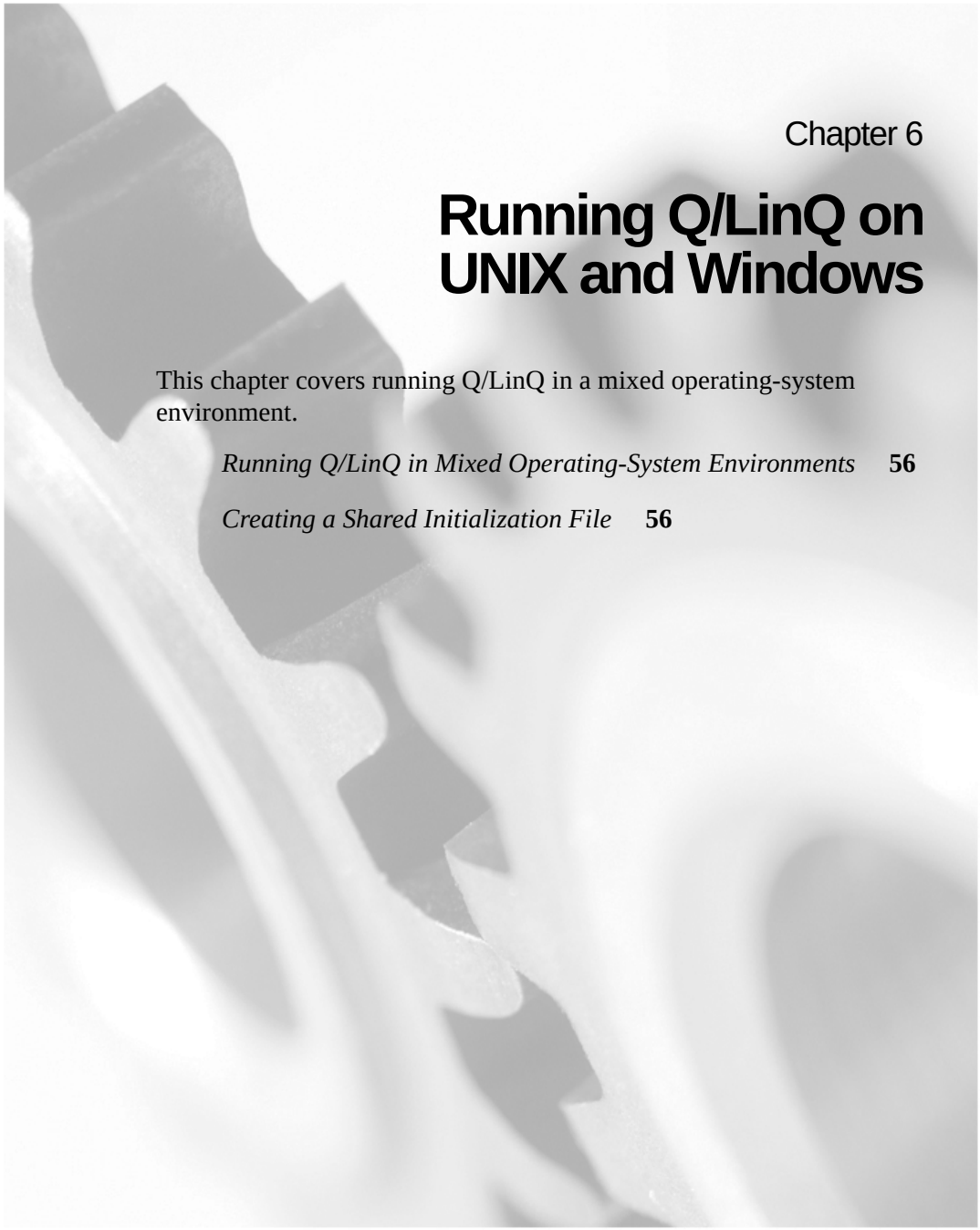
Installation Reference

This section includes the following installation reference material.

Running Q/LinQ on UNIX and Windows **55**

Initialization Files and Installation Directories **59**

UNIX System Upgrades **69**



Chapter 6

Running Q/LinQ on UNIX and Windows

This chapter covers running Q/LinQ in a mixed operating-system environment.

Running Q/LinQ in Mixed Operating-System Environments **56**

Creating a Shared Initialization File **56**

Running Q/LinQ in Mixed Operating-System Environments

Q/LinQ can be run simultaneously on both UNIX and Windows operating systems. To run Q/LinQ in a mixed operating-system environment, use one of the following configuration options:

- Option One: Set up a configuration in which all Q/LinQ clients reference separate `.ini` files for each operating system. That is, UNIX clients reference UNIX-specific `qqapi.ini` files and Windows clients reference Windows-specific `qqapi.ini` files.
- Option Two: Set up a configuration where Q/LinQ clients on Windows systems and Q/LinQ clients on UNIX systems reference the same `qqapi.ini` file on a UNIX Q/LinQ server. This configuration is covered in the following section.

Creating a Shared Initialization File

1. Use the instructions in Chapter 2 or Chapter 4 to set up a Q/LinQ server and at least one client, both of the same operating system.

2. Use file-sharing software to provide clients that have a different operating system from that of the server you set up in step 1 with access to the directory containing `qqapi.ini`.

   If you installed a Windows server and Windows client, `qqapi.ini` is located in the MFG/PRO working directory on the client.

   Note You may want to copy this file to the UNIX server before modifying it for use by UNIX clients.

   If you installed a UNIX server and client, `qqapi.ini` is located in the Q/LinQ installation directory.

   Example For a Windows client, map a network drive to the directory on the UNIX server that contains `qqapi.ini`. This file was created during the UNIX Q/LinQ installation process.

3. Edit `qqapi.ini` to contain information for clients with operating systems that differ from that of the original Q/LinQ client.

See “Q/LinQ Initialization Files” on page 60.

Example For Windows clients, edit the `[win32]` section of `qqapi.ini` on the UNIX server to contain Windows client-specific information.

4 For each Q/LinQ client on all systems, modify the PROPATH in the MFG/PRO `.ini` file to contain the directory path to the modified `qqapi.ini`.

For UNIX clients, follow the instructions in “Modifying the PROPATH” on page 19. For Windows clients, refer to “Modifying the Client PROPATH (Optional)” on page 39.

Chapter 7

Initialization Files and Installation Directories

This chapter covers the Q/LinQ files and directories created during the installation process.

Q/LinQ Initialization Files **60**

Q/LinQ Installation Directories **64**

Q/LinQ Initialization Files

Installation creates several initialization files (`.ini`). These files contain necessary startup information for Q/LinQ and the Q/LinQ message-oriented middleware (MOM) adapter. Although the data in these files are automatically generated during the installation process, you may need to edit them if applicable parameter values or file locations change.

The following sections provide brief descriptions of key variables in each `.ini` file.

Q/LinQ Initialization File

The Q/LinQ initialization file, `qqapi.ini`, is created during the Q/LinQ installation process. This file indicates the paths to the directories containing the non-Progress resource files used by Q/LinQ.

Important The path to `qqapi.ini` *must* be included in the PROPATH for each Q/LinQ client. For details, refer to the appropriate section for your installation environment:

- For UNIX servers with UNIX clients, see “Modifying the PROPATH” on page 19.
- For Windows servers with Windows clients, the location of `qqapi.ini` is automatically included in each Q/LinQ client PROPATH during the client installation procedure. See “Running the Client Installation” on page 37.
- For mixed operating-system environments, see “Creating a Shared Initialization File” on page 56.

Each entry in `qqapi.ini` is comprised of an inactive line that briefly describes the applicable directory’s function, followed by a line containing a directory variable, an equal sign, and a directory path.

Example The following `.ini` file entry indicates the directory containing the Q/LinQ mapping specification files on a Windows server:

```
;Directory for mapping specification files

qqMapSpecDir=C:\Program Files
   \QAD\QlinqNTServer\resource\mapspec\
```

Example The following `.ini` file entry indicates the directory containing the Q/LinQ mapping specification files on a UNIX server:

```
;Directory for mapping specification files
qqMapSpecDir=/qad/mfgpro/90/testobj/resource/mapspec/
```

`qqapi.ini` is divided into the following major sections:

- The `[unix]`section contains Q/LinQ directory paths for UNIX servers and clients.
- The `[win32]` section contains Q/LinQ directory paths for Windows servers and Windows clients.

Since `qqapi.ini` is divided into sections for specific operating systems, you can use the same initialization file for both Windows and UNIX clients.

See Chapter 6, “Running Q/LinQ on UNIX and Windows,” on page 55.

Q/LinQ clients must have write access to the directories specified for the following `qqapi.ini` directory variables. To avoid file contention, the directory specified for each of these variables must be unique:

- `qqWorkDir`
- `qqArchiveDir`
- `qqEditDir`
- `qqExpLog`
- `qqImpLog`

Q/LinQ MOM Adapter Initialization File

The Q/LinQ MOM adapter initialization file, `QqMomAdapter.ini`, is created during the Q/LinQ installation process and is stored on the Q/LinQ server in the `mom` directory. This file contains the runtime parameters and the MQSeries connection parameters for the Q/LinQ MOM adapter.

`QqMomAdapter.ini` is divided into the following major sections:

- The `[startup]`section contains runtime parameters for the Q/LinQ MOM adapter.
- The `[MQSeries]` section contains the connection variables for the MQSeries MOM.

- The `[inbound routing]` section maps MQSeries queue names to Q/LinQ external application IDs when importing from MQSeries into Q/LinQ.
- The `[outbound routing]` section specifies the MQSeries queue to which Q/LinQ documents should be sent when exporting from Q/LinQ to MQSeries.

See “Registering Multiple Applications” on page 112 for additional details.

The initialization file provided with the Q/LinQ installation has additional information and examples on populating the two routing sections.

Important While Q/LinQ application ID values in the Progress environment are case-insensitive, the MOM adapter and Java environment in general use case-sensitive processing. Therefore, the case of the application ID in the `QqMomAdapter.ini` file must agree with the case used in the Q/LinQ menu screens.

The adapter reads the initialization file once at startup. If the initialization file is modified after the adapter is running, the adapter will reread it when the next Q/LinQ session connects. Earlier sessions will continue to run with the old values. Table 9.1 indicates which parameters may not be modified once the adapter has started.

Table 7.1 and Table 7.2 provide brief descriptions of the variables in each section of `QqMomAdapter.ini`. Variables are listed in the order in which they occur in the `.ini` file.

Table 7.1 Q/LinQ MOM Adapter `.ini` File, `[startup]` Section

Variable	Description
MomType	The MOM product with which the Q/LinQ MOM adapter is used. This variable cannot be modified after startup of the adapter. **Note:** Q/LinQ supports only the MQSeries MOM from IBM. **Valid Entry:** MQSeries
MomTimeOut	The time (in seconds) that the adapter should wait for a response from the MOM when reading data from it. **Valid Entry:** Any positive integer. If set to 0 (zero), it waits indefinitely.
QlinqTimeOut	The time (in seconds) that this adapter should wait for data to be sent from Q/LinQ. **Valid Entry:** Any positive integer. If set to 0 (zero), it waits indefinitely.

Variable	Description
MomPollTime	The optional MOM polling interval (in seconds). This is the time the adapter waits between requests to the MOM for more documents, once it finds no documents remaining to import. If not specified, it defaults to 1 second. **Valid Entry:** Any positive integer.
Port	The port number on the MOM adapter host machine through which the adapter should communicate with Q/LinQ. This is not the same as the port parameter in the MQSeries selection. This variable cannot be modified after startup of the adapter.
LogFile	The name of the log file generated by the Q/LinQ MOM adapter. This variable cannot be modified after startup of the adapter.
LogFileOverwrite	Indicates whether existing Q/LinQ MOM adapter log entries are overwritten by new entries. This variable cannot be modified after startup of the adapter. **Valid Entries:** • true: Existing entries are overwritten by new entries. • false: New entries are appended to the end of the log.
LogDestination	The destination of Q/LinQ MOM adapter log entries. **Valid Entries:** • file: Log entries are placed in the file indicated in the LogFile variable. • display: Log entries are displayed to the terminal and are not saved to a file. • both: Log entries are displayed to the terminal and are placed in the file indicated in the LogFile variable.
LogMode	Indicates the type of Q/LinQ MOM adapter log entries generated. **Valid Entries:** • silent: No log entries are generated. • verbose: Log entries are generated only for critical information such as warning and error messages. • debug: Detailed log entries are generated for all transactions. **Note:** Debug mode is not recommend for normal use, as it creates a large log file and slows performance.

Variable	Description
QlinqConnectSleep	The time (in seconds) that the adapter waits when executing a Q/LinQ connection retry. This parameter is provided only for troubleshooting purposes. Unless there is a problem, do not assign a value to it. **Valid Entry:** Any positive integer.
MomGetTimeOut	The time (in milliseconds) that the MOM waits to get a single message from the queue before returning control to the adapter. This parameter is provided only for troubleshooting purposes. Unless there is a problem, do not assign a value to it. **Valid Entry:** Any positive integer.

Table 7.2 Q/LinQ MOM Adapter `.ini` File, `[MQSeries]` Section

Variable	Description
host	The IP address or machine name of the MQSeries server.
port	The port number of the MQSeries channel for the Q/LinQ MOM adapter.
QManager	The name of the MQSeries queue manager for the Q/LinQ MOM adapter.
channel	The name of the MQSeries channel for the Q/LinQ MOM adapter.

Q/LinQ Installation Directories

The following sections list the directories and files created during the Q/LinQ installation process.

Note The forward slashes (/) shown in these examples are UNIX-specific. Windows systems use backward slashes (\) to separate directory levels in path statements.

Version

The `version` file is located in *`QlinqInstallDir`* and contains a single line identifying the Q/LinQ version. Use this file for reference when calling QAD's Global Support hot line.

Important Do not modify this file.

Archive

The `archive` directory stores archived Q/LinQ history. This directory remains empty until Export/Import Doc Delete/Archive (36.8.23) is run. For clients to run the Export/Import Doc Delete/Archive function, they must be able to write to the `archive` directory.

`qqapi.ini`, created during the installation process, sets the `qqArchiveDir` parameter to the following path:

```
QlinqServerInstallDir/resource/archive
```

Bin

The `bin` directory contains a subdirectory for each of the Q/LinQ-supported operating systems. Each system-specific subdirectory contains all of the pre-compiled executables required by Q/LinQ to handle data communications. These subdirectories also contain the `qqsend` and `qqrecv` testing utilities. Use these utilities to test the Q/LinQ TCP/IP sockets-based data communication facilities.

`qqapi.ini`, created during the installation process, references the applicable `bin` subdirectory for your operating system in the `qqExecDir` parameter. The following example is for a Windows client:

```
qqExecDir=C:\Program Files
   \QAD\QlinqServerInstallDir\resource\bin\nt40int\
```

Classes

The `classes` directory contains the pre-compiled Java classes used by Q/LinQ, including several `.jar` files that implement the Q/LinQ messaging API.

This directory also contains `qqtest.jar`. This file contains the `QqSend` and `QqRecv` testing utilities. Use these utilities to test the Q/LinQ TCP/IP sockets-based data communication facilities.

See “Testing Communication” on page 122.

If the MQSeries adapter is used or any development is done using the Java version of the Q/LinQ messaging API, this directory and each of the `.jar` files in it must be included in the `CLASSPATH`. In a UNIX

environment, the `CLASSPATH` is located on the Q/LinQ server. In a Windows environment, the `CLASSPATH` is located on the Q/LinQ Windows clients.

Data

The `data` directory contains two Extensible Markup Language (XML) Document Type Definition (DTD) files that describe all of the QAD-defined extensions to the Open Application Group (OAG) Business Object Documents (BODs) used by Q/LinQ: `qad_mfgpro_90.dtd for` MFG/PRO 9.0 and `qad_mfgpro_eb.dtd` for MFG/PRO eB and eB2. This file provides the information required to construct a composite DTD in multi-vendor environments where users or software vendors are defining extensions to the same BODs as QAD.

The `data` directory also contains miscellaneous Q/LinQ and Progress data files used for demonstration and training purposes.

Explog

The `explog` directory stores maintenance log files for exported documents. This directory is initially empty; Reload Edited Export/ Import Doc (36.8.14) writes files to it. For clients to run the Reload Edited Export/Import Doc function, they must be able to write to the `explog` directory.

`qqapi.ini`, created during the installation process, sets the `qqExpLog` parameter to the following path:

`QlinqServerInstallDir``/resource/explog`

Implog

The `implog` directory stores maintenance log files for imported documents. This directory is initially empty; Reload Edited Export/ Import Doc (36.8.14) writes files to it. For clients to run the Reload Edited Export/Import Doc function, they must be able to write to the `implog` directory.

`qqapi.ini`, created during the installation process, sets the `qqImpLog` parameter to the following path:

`QlinqServerInstallDir``/resource/implog`

Mapspec

The `mapspec` directory contains all of the XSLT stylesheets provided with Q/LinQ for mapping XML-based BODs.

`qqapi.ini`, created during the installation process, references the `mapspec` directory in the `qqMapSpecDir` parameter.

MOM

The `mom` directory contains the Q/LinQ message-oriented middleware (MOM) adapter, `QqMomAdapter`. MQSeries from IBM is the only MOM supported by this adapter. The `mom` directory also contains `QqMomAdapter.ini` and the Java `.class` and `.jar` files.

If MQSeries is used, this directory and each of its files must be included in the `CLASSPATH`. In a UNIX environment, the `CLASSPATH` is located on the Q/LinQ server. In a Windows environment, the `CLASSPATH` is located on the Q/LinQ Windows clients.

Scripts

The `scripts` subdirectory contains sample batch scripts (`.bat`) for running the following key Q/LinQ programs:

- Send Export Documents (36.8.7)
- Receive Import Documents (36.8.9)
- Process Import Documents (36.8.10)
- Export/Import Doc Delete/Archive (36.8.23)

For UNIX systems, use these script files and the `.ksh` file extension to build UNIX `cron` scripts that run common Q/LinQ functions.

For Windows systems, refer to `qqbatch.txt` for information on implementing this functionality.

Note In Windows environments, only MFG/PRO Windows character clients can run `.bat` files.

Work

The `work` directory stores work files required by mapping utilities or user-written programs called by Q/LinQ. It also stores temporary text files containing editable Q/LinQ document data created by Dump Export/ Import Doc for Edit (36.8.13).

`qqapi.ini`, created during the installation process, sets the `qqEditDir` and `qqWorkDir` parameters to the following path:

`QlinqServerInstallDir``/resource/work`

Lib

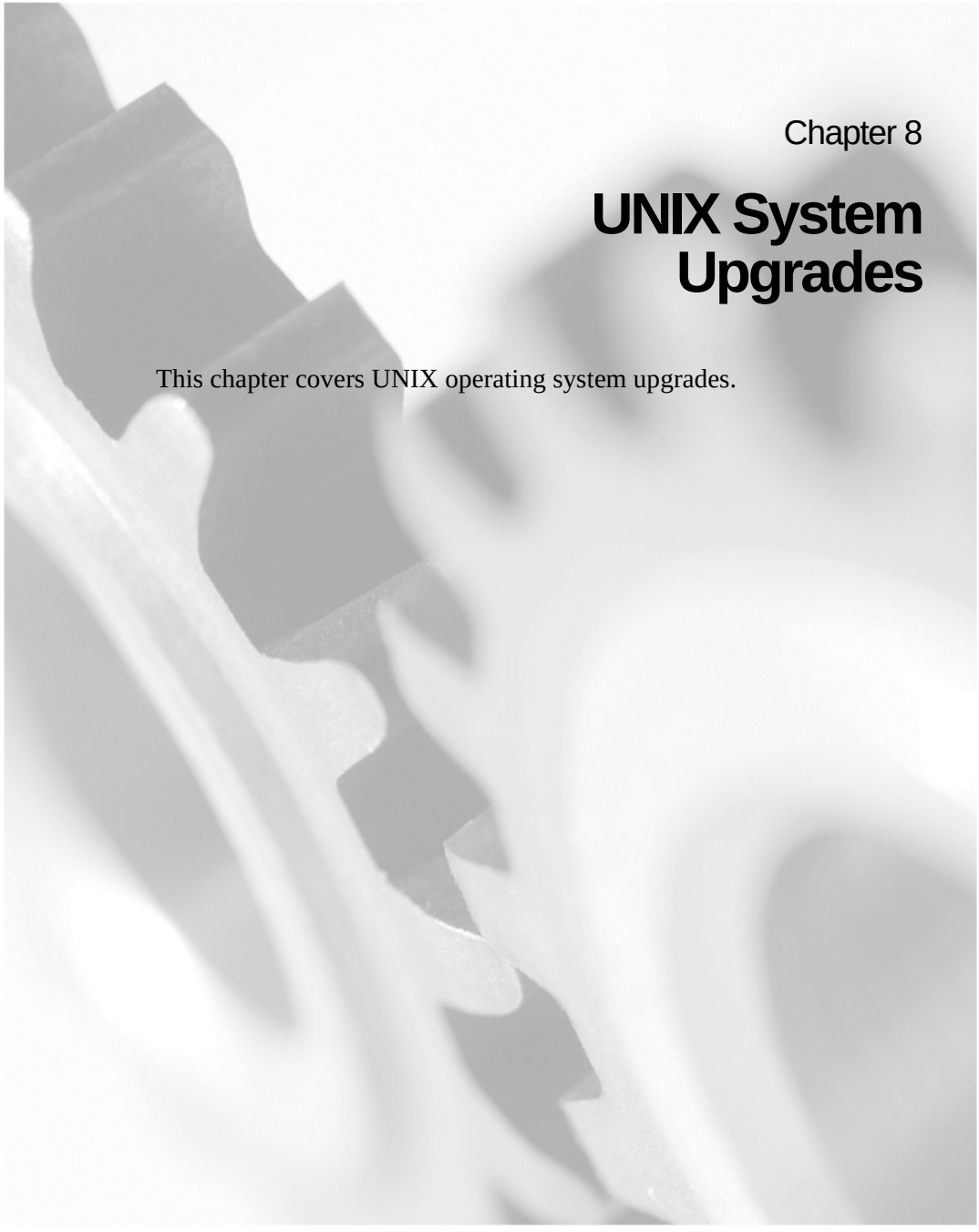
The `lib` directory contains subdirectories for each operating system, with the C static library files required to build connectors or adapters using the Q/LinQ messaging API.

The `docs` subdirectory contains `javadoc` HTML files describing the class interfaces that make up the Java version of the messaging API.

Samples

The `samples` directory contains subdirectories with source-code samples for each of the Q/LinQ APIs: publishing, mapping, communication (messaging, stream), and processing. These sample programs are used in the Q/LinQ developer training offered by QAD and are a useful reference for software developers using the Q/LinQ APIs.

This directory also contains sample UNIX shell scripts for running the key Q/LinQ functions in batch mode; for example, through UNIX `cron`.



Chapter 8

UNIX System Upgrades

This chapter covers UNIX operating system upgrades.

For qqapi.ini details, see “Q/LinQ Initialization File” on page 60.

If you upgrade the operating system version, you may be able to continue to use the same Q/LinQ code by modifying qqapi.ini.

Use the following instructions to modify qqapi.ini to run HP-UX versions higher than 11.00 and Solaris versions higher than 2.6.

Note QAD cannot guarantee that Q/LinQ code compiled for a specific operating system version will operate on subsequent versions.

1. Use the following command to determine the operating system of the machine on which you want to run Q/LinQ.

   ```
   uname -rs
   ```

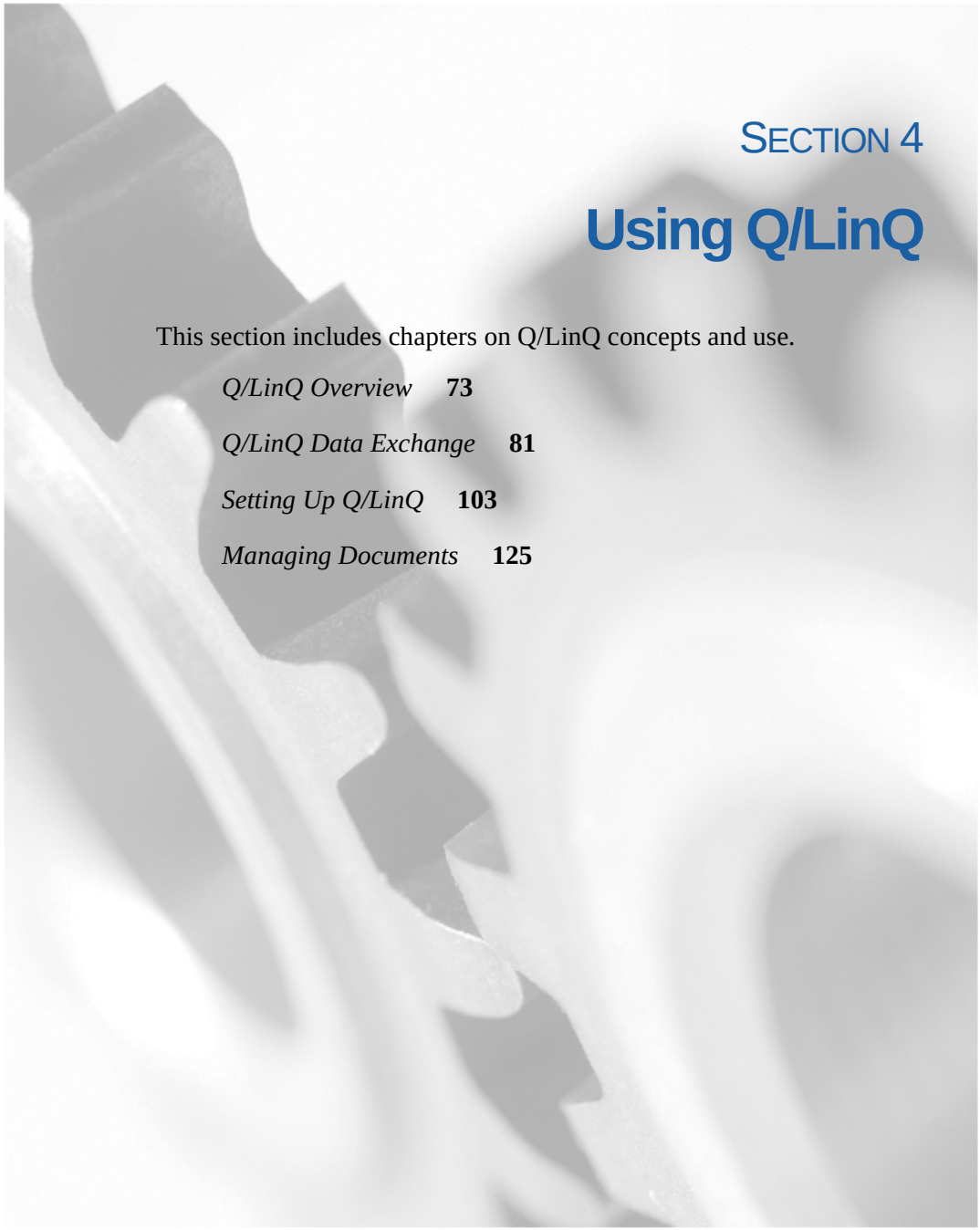
   This command provides the name and version of the operating system; for example, HP-UX B.12.30. Record this information for use in step 3.

2. From the Q/LinQ installation directory, open qqapi.ini in a text editor.

3. On the O/S-specific subdirectory maps line, change the operating system version to the version displayed by the uname -rs command entered in step 1. Then, enter the name of the Q/LinQ subdirectory that corresponds to the highest version of your operating system supported by Q/LinQ.

Example Enter the operating system version—HP-UX B.12.30—to the left of the equal sign and the appropriate Q/LinQ OS-specific subdirectory for that operating system—hpux1010—to the right of the equal sign:

```
; O/S-specific subdirectory maps
   HP-UX B.12.30=hpux1010
```



SECTION 4

Using Q/LinQ

This section includes chapters on Q/LinQ concepts and use.

Q/LinQ Overview **73**

Q/LinQ Data Exchange **81**

Setting Up Q/LinQ **103**

Managing Documents **125**

Chapter 9

Q/LinQ Overview

This chapter introduces Q/LinQ and provides a broad description of Q/LinQ architecture and the import/export process. It also lists Q/LinQ menu programs.

Introduction **74**

Q/LinQ Features **75**

Q/LinQ Interface Architecture **76**

Data Flow **78**

Q/LinQ Menu **79**

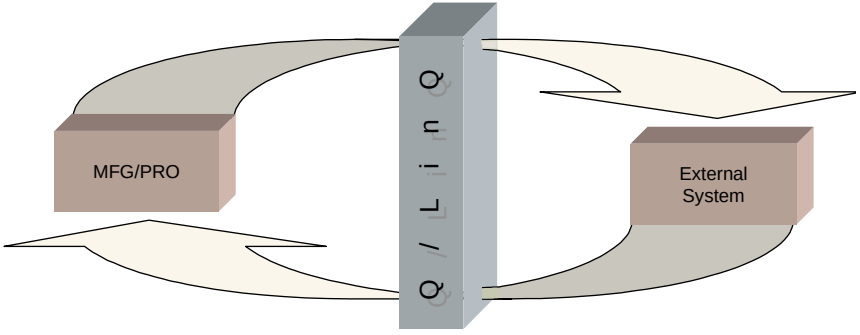
Introduction

Q/LinQ is a complete tool set for building integrations with third-party or in-house applications for complex data exchange. It is also the infrastructure for administering and managing the data exchange between MFG/PRO and these external applications.

These external applications include logistics and distribution systems, factory automation systems, scheduling systems, and sales-order-entry applications.

Q/LinQ is a framework of interfaces for data coming into or going out of MFG/PRO (see Figure 9.1). Q/LinQ receives the data and passes it to its destination. Data can come from a file, an MFG/PRO process, or an application or system external to MFG/PRO. The destination can be a file, another MFG/PRO process, or an application or system external to MFG/PRO. Q/LinQ can instruct the external application to process the data and then inform Q/LinQ if the processing succeeded, failed, or generated errors.

Fig. 9.1
Q/LinQ Data Flow



This chapter describes the Q/LinQ architecture and data flow and concludes with the Q/LinQ menu list. This chapter also provides useful information for:

- MFG/PRO users who initiate or monitor data exchange between MFG/PRO and other applications
- System administrators who set up Q/LinQ
- Programmers who implement application program interfaces (APIs)

After reviewing the information in this chapter, see the following:

- Chapter 10, “Q/LinQ Data Exchange,” discusses the ways Q/LinQ exchanges data, the form of the data, and the APIs that support communication with external applications.
- Chapter 11, “Setting Up Q/LinQ,” gives system administrators the information they need to manage Q/LinQ’s Controls, enter Q/LinQ’s default settings, and maintain communication processes.
- Chapter 12, “Managing Documents,” provides instructions for importing and exporting data.
- *Technical Reference: Q/LinQ* provides language-specific API specifications, including the library of routines, UNIX and Windows include files, Java classes, parameters, and template APIs that programmers require to build custom APIs to external applications.

Q/LinQ Features

The Q/LinQ features include user-interface (UI) emulation, document management, error recovery, and notification:

- The Q/LinQ UI emulation improves on CIM functionality; it processes data significantly faster than the CIM Interface. Q/LinQ accepts, without modification, files that use CIM Interface tags.
- Support for multiple concurrent import and export sessions.
- Ability to query the status and contents of the transaction and error files, to edit or rerun transactions with errors, and to archive or delete files according to specified criteria.
- Ability to restart the data-exchange process from the last successfully completed step if processing is interrupted.
- E-mail notification regarding the success or failure of transaction processing.

The location of information on the CIM Interface is described on page 2.

Q/LinQ includes generic API frameworks and the technical information required to build custom integrations with any external application. The *Technical Reference: Q/LinQ* includes detailed information about API parameters and functions.

Tip
QAD’s Global Services builds custom APIs.

Q/LinQ supports building custom adapter programs for:

- Communicating between MFG/PRO and any external application
- Mapping data between MFG/PRO and external application formats
- Publishing data to Q/LinQ from MFG/PRO applications
- Processing data and directly accessing the MFG/PRO database

Q/LinQ Interface Architecture

This section covers the components of Q/LinQ's interface for direct data exchange between MFG/PRO and external applications—in addition to its basic capability of UI emulation.

An interface makes a boundary between two systems permeable to data and services. An API implements this permeable boundary through a set of routines, protocols, and tools for building software. The Q/LinQ interface is a set of APIs that allows applications of disparate formats and languages to exchange data and services that support the applications.

Q/LinQ's generic APIs are the framework for custom interfaces. Custom interfaces can be written in Progress, C++, C, or Java. C programs must execute on UNIX or Windows operating systems.

Through custom interfaces, Q/LinQ has the potential to communicate with:

- Non-QAD applications
- Message-oriented middleware (MOM) or existing in-house infrastructure that fulfills middleware functions
- Other MFG/PRO installations or QAD applications

The more powerful and flexible a custom interface, the more components it requires. Interface components can include any of the following programs:

- Adapter programs mediate differences in the protocols that applications use to communicate, as well as differences in the way applications are invoked remotely. Adapters provide a set of entry points into and calls out of an application. They mediate between the layer of one application exposed and supported for outside access and the similar layer in a different application.

- Middleware is a type of connectivity software for complex client/server systems that support distributed runtime processing and application development environments. Q/LinQ supports MQSeries, the message-oriented middleware from IBM.
- Data mappers are programs or specification files used to transform data from one format into another. Q/LinQ provides a set of maps based on Extended Stylesheet Language Transformation (XSLT) technology.
- Gateways are programs that link different types of networks or applications. In MFG/PRO, gateway programs enable direct access to the database. They process data received from an external application or extract data to send to an external application.

Figure 9.2 shows the connectivity capability of Q/LinQ when used to its full extent, although not all of the possible interface components are shown.

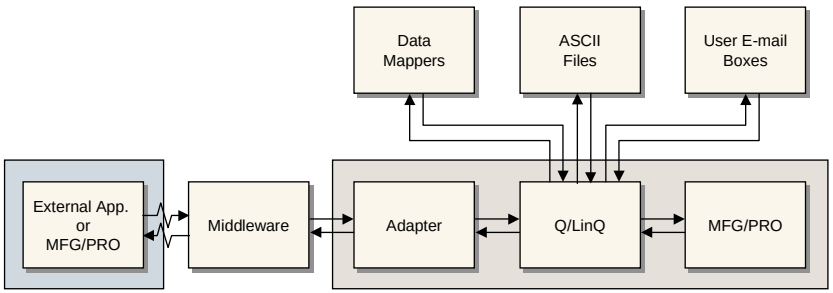
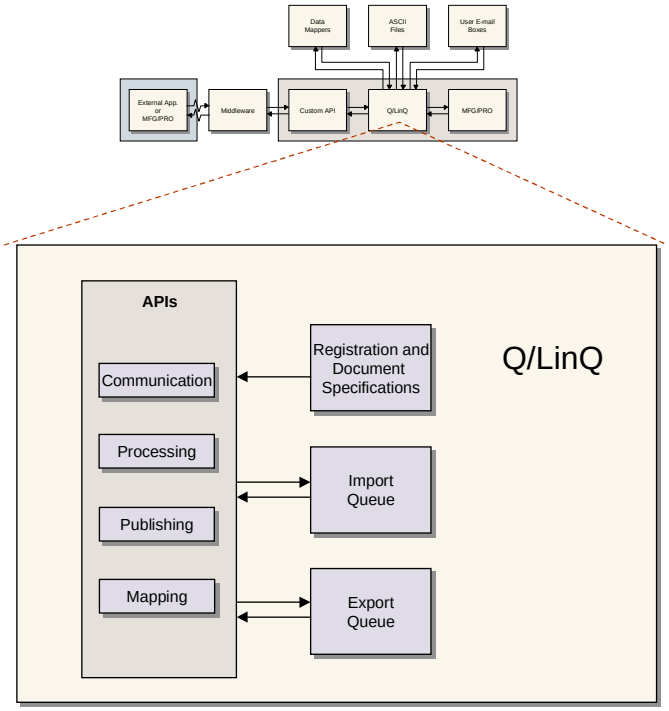


Fig. 9.2 Interface Architecture

Figure 9.3 shows the internal Q/LinQ architecture in relation to the overall interface architecture shown in Figure 9.2.

Fig. 9.3
Q/LinQ System Architecture



Data Flow

Figure 9.4 shows the data flows among the APIs within Q/LinQ that enable direct data exchange between MFG/PRO and external applications.

The data flow for exporting from MFG/PRO to an external application follows these steps:

1. A user-written Progress program acquires the transaction data from MFG/PRO and calls the publishing API.
2. The publishing API posts the data to the export data queue.
3. The mapping API transforms the format of the data to a format the external application can read.
4. The communication API sends the data to the external application.

The data flow for importing into MFG/PRO from an external application follows steps similar to the export process, but in reverse.

1 The communication API receives the data from the external source system and posts it to the import queue.

2 The mapping API transforms the format of the external application data to a format MFG/PRO can read.

3 The processing API passes the data to MFG/PRO for processing.

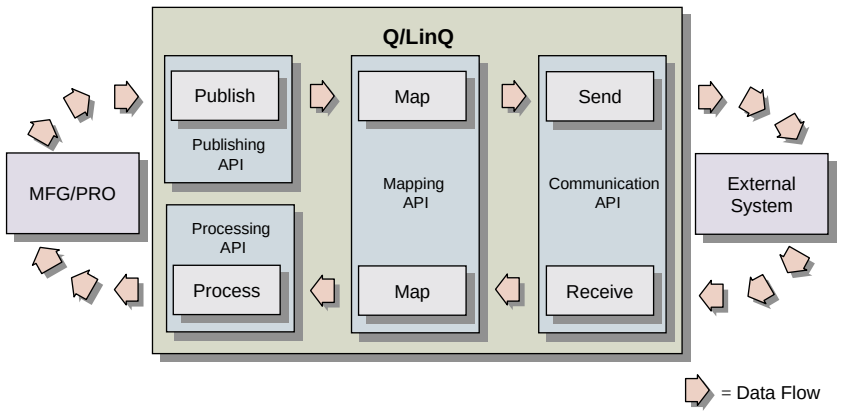


Fig. 9.4 Export and Import Data Flow Between MFG/PRO and an External Application

Q/LinQ Menu

Table 9.1 lists the programs that are used to set up Q/LinQ, exchange data, and edit documents. The Q/LinQ menu structure differs by MFG/PRO version.

When referencing a program with two menu locations, both locations are listed; for example Export Specification Maint (36.8.1.2; 36.8.2 in 9.0).

Table 9.1 Q/LinQ Menu (36.8)

MFG/PRO Menu Number		Description	Program
eB and eB2	**9.0**		
36.8.1		Q/LinQ Setup Menu	
36.8.1.1	36.8.1	Register External Application	qqaprmt.p
36.8.1.2	36.8.2	Export Specification Maint	qqespmt.p
36.8.1.3	36.8.3	Import Specification Maint	qqispmt.p

MFG/PRO Menu Number		Description	Program
eB and eB2	**9.0**		
36.8.1.20	Not available	Code Mapping Maintenance	qqcodmt.p
36.8.1.21	Not available	Code Mapping Inquiry	qqcodiq.p
36.8.5	Same	Interface Session Monitor	qqsesmon.p
36.8.7	Same	Send Export Documents	qqsnd.p
36.8.9	Same	Receive Import Documents	qqrcv.p
36.8.10	Same	Process Import Documents	qqimprc.p
36.8.11	Same	Debug CIM Document	qqimdbug.p
36.8.12	Same	Process Received Acknowledgment	qqprcack.p
36.8.13	Same	Dump Export/Import Doc for Edit	qqdp.p
36.8.14	Same	Reload Edited Export/Import Doc	qqld.p
36.8.15	Same	Document Control Tag Maint	qqtagmt.p
36.8.16	Same	Export/Import Document Query	qqbr.p
36.8.17	Same	Export/Import Document Report	qqierp.p
36.8.18	Same	Dump Export/Import Docs to File	qqwrt.p
36.8.23	Same	Export/Import Doc Delete/Archive	qqrup.p
36.8.24	Same	Q/LinQ Control	qqctrmnt.p

Chapter 10

Q/LinQ Data Exchange

This chapter describes the asynchronous processes Q/LinQ uses to exchange documents with external applications. It includes information on how Q/LinQ uses application program interfaces (APIs) to transform data and transfer it to the receiving system, how it retains document sequences, and how it sends and receives notifications of document processing.

Interface Message Standards **82**

Asynchronous Interface **82**

Transaction Document Groups **84**

Autoacknowledgments and Notifications **87**

Exchanging Documents **90**

Q/LinQ APIs **92**

Interface Message Standards

Q/LinQ exports or imports data messages in the form of electronic *documents.* A document is the business data pertinent to a single instance of a business process. For example, the output of an MFG/PRO transaction such as a purchase order entry comprises a document. A single document is the content of an interface message.

To support interoperability among applications, Q/LinQ supports the use of the Open Applications Group (OAG) standards for business data communications: the Open Applications Group Integration Specification (OAGIS).

Q/LinQ document are OAGIS Business Object Documents (BODs). OAGIS BODs can be expressed in XML (extensible markup language). XML supports the definition of data types and parameters and is particularly suited to presenting and transporting data over the Internet.

Note Visit the OAG Web site, http://www.openapplications.org, for more information about OAGIS and implementation of XML BODs.

Asynchronous Interface

BOOI Messages

Q/LinQ can exchange documents, or interface messages, continuously as a batch of one immediate (BOOI). BOOI messages are created and sent immediately as a result of application events, rather than exported later in batches. Q/LinQ receives the BOOI messages as MFG/PRO or the external application creates and passes them. Q/LinQ queues and sends the messages to the external application or receives and queues the messages for processing by MFG/PRO.

Messages are not *synchronous*, that is, received or processed at the destination at the same time they were created. Synchronous processing must wait for a reply from the destination, often slowing performance in the sending application. Q/LinQ does not support synchronous processing and real-time response.

Q/LinQ employs *asynchronous* data processing. In contrast to a synchronous process, an asynchronous process does need not to complete

for the originating event to complete. Further, asynchronous processing does not necessarily take place soon after the initiation of the originating event. Q/LinQ queues the data in tables until the receiving application accepts it for processing, and then the process completes.

Multi-Tier Environments and MOM

Asynchronous, queue-based messaging supports data exchange in multi-tier, distributed environments. It does not require an active communication link during the transaction; it can run in the background, requiring no user input.

When communicating through message-oriented middleware (MOM) adapters, Q/LinQ can use any publish-subscribe mechanisms supported by the MOM API. In *publish-subscribe messaging*, an application publishes documents, or makes them available to an unknown, and possibly changing, list of subscribers. The other application subscribes to or requests receipt of the documents. This messaging mechanism is suited for data synchronization and event notification.

See “Message-Oriented Middleware” and “MOM Adapter” on page 101.

Brokering communications between multiple publishers and multiple subscribers usually requires middleware such as an object request broker (ORB) or MOM. The gateway programs and adapter in Figure 10.1 are custom written specifically for the external application, MOM, or document.

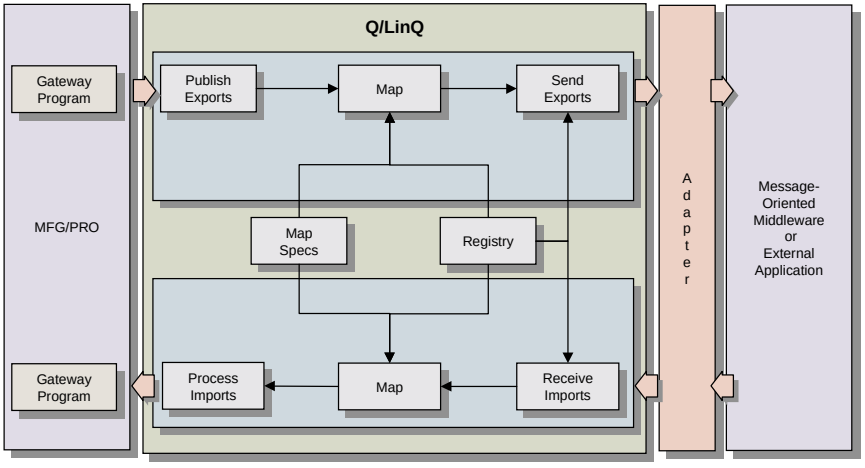


Fig. 10.1
Data Flow Through Q/LinQ

If the external application uses direct messaging and has robust APIs that parallel the Q/LinQ APIs, MOM might not be necessary. Otherwise, MOM is necessary when sending and receiving applications are not directly aware of each other (anonymous publish-subscribe). In this case, the MOM ensures that published messages are routed to their appropriate destinations.

Transaction Document Groups

A Q/LinQ *session* is a single executing Q/LinQ menu procedure that is publishing, sending, receiving, processing, archiving, or deleting documents. A Q/LinQ session can import or export one document or a group of documents in a single file or from an executing process.

To maintain data integrity, some types of documents must be processed all together or in the sequence in which they were created or imported. An example of this is a series of shop floor transactions, such as issues, completions, or receipts for a single work order.

This section covers the following topics related to document groups:

- Document group types
- Document group assignments
- Group processing example

Document Group Types

To accommodate the business need for document sequencing, Q/LinQ recognizes groups of documents defined as a series or a batch:

- In a series group, documents must be mapped and processed in the same order that Q/LinQ receives them. If a document in a series group contains an error that prevents Q/LinQ from processing it, Q/LinQ does not process the documents that follow the defective one. However, it does commit to the database any documents that it has already successfully processed.
- In a batch group, documents must also be processed in sequential order. However, Q/LinQ only commits a document if *all* documents in the batch are processed successfully. When errors occur, Q/LinQ rolls back any processed transactions.

If a document is neither series nor batch, Q/LinQ processes it as a group of one. Q/LinQ can process and commit these individual-document groups in any order relative to other documents.

Document Group Assignments

Q/LinQ assigns documents in a file to groups according to the value of the `@@GROUP` and `@@GROUPEND` tags.

See Appendix B, “Text File Control Tags,” on page 167.

Output documents from an executing process are assigned to groups within the API written by the programmer.

Q/LinQ assigns unique IDs to import and export documents, document groups, and sessions:

- The document ID is the primary means of locating the document data and log record.
- In addition to the document ID, Q/LinQ gives each group an ID.
- When Q/LinQ establishes communications to import from or export to an external application, it assigns a unique session ID to the connection.

Q/LinQ displays these IDs in reports and browses. The IDs also display on the Interface Session Monitor (36.8.5) screen. They are useful for troubleshooting document processing and communication errors.

See “Monitoring Q/LinQ Sessions” on page 145.

The session ID acts as a lock on a document, preventing multiple Q/LinQ sessions from simultaneously exchanging that document with an external application. However, one Q/LinQ session can receive imports from an external application while another Q/LinQ session concurrently sends exports to the same external application.

Note Limit the use of groups for imported data. Q/LinQ does not release the locks on the documents until they have all been processed or rolled back. This feature could cause database-contention problems for other MFG/PRO users.

Group Processing Example

Table 10.1 lists example entries in the import queue. Figure 10.2 graphically describes the relationships of the documents, groups, and sessions.

A user opened Process Import Documents (36.8.10) to process documents 1001, 1002, 1003, and 1004. Q/LinQ gave that session the ID 1001. The user then opened Process Import Documents again, which Q/LinQ defined as session 1002, to process documents 1005, 1006, 1007, 1008, and 1009.

Table 10.1
Sample Entries in Import Queue

Document ID	Group Type	Group ID	Session ID
1001	series	0010	1001
1002	series	0010	1001
1003	series	0010	1001
1004	series	0010	1001
1005	individual	0011	1002
1006	batch	0013	1002
1007	batch	0013	1002
1008	batch	0013	1002
1009	individual	0012	1002

During processing of sessions 1001 and 1002, if only document 1007 contains an error, Q/LinQ completes these steps:

1. Processes and commits documents 1001–1005.
2. Processes and then rolls back documents 1006 and 1007.
3. Skips document 1008.
4. Processes and commits document 1009.

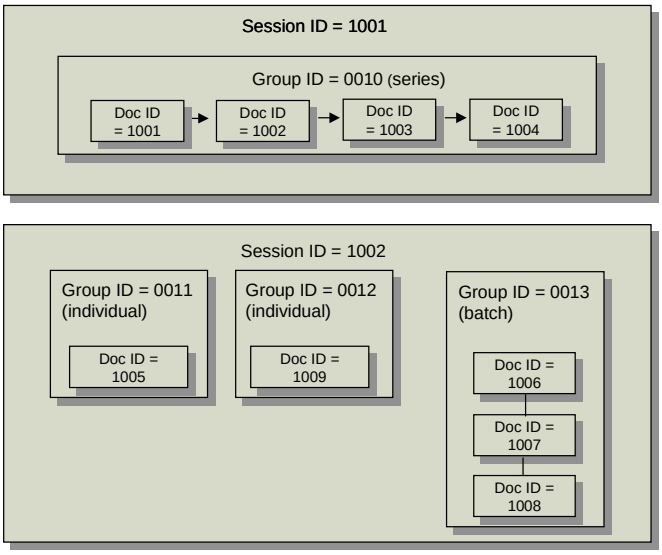


Fig. 10.2
Batch, Series, and Individual Groups

Autoacknowledgments and Notifications

Q/LinQ can send to or request from the external application an automatic acknowledgment regarding the results of document processing. Based on Control settings, Q/LinQ also can notify an MFG/PRO user by e-mail of processing events.

See “Exchanging Documents” on page 90.

Note This autoacknowledgment has no relation to the communications acknowledgment of data packets.

See *Technical Reference: Q/LinQ*.

This section covers the following topics:

- Autoacknowledgment messages
- Acknowledgment generation and level
- Acknowledgment processing
- E-mail notification

Autoacknowledgment Messages

In addition to results of processing, an autoacknowledgment can include the following:

- Any exception or information messages generated during processing
- Information about the sender of the document
- The date and time of document creation

Autoacknowledgment can help synchronize activity with the external application by providing automatic, event-driven notification messages.

For import documents, Q/LinQ generates and sends a message to the external source application after Q/LinQ processes the imported documents. Q/LinQ queues, maps, and exports the message in the same manner as other export documents. The record of the acknowledgment in the Q/LinQ import log includes a date and time stamp.

See “One-to-One Mapping” on page 98.

When an import document is mapped into multiple documents, its autoacknowledgment includes processing results for all the documents. That is, Q/LinQ generates no more than one autoacknowledgment per import document.

See page 89.

For export documents, Q/LinQ can ask an external application that receives its documents to send back an autoacknowledgment message. This import document informs Q/LinQ of the results of the processing of the previously exported document by the external application. The record of the acknowledgment in the export log includes a date and time stamp. Q/LinQ imports the message the same way it imports other documents: it is received, mapped, and processed.

Autoacknowledgment Generation

See“Q/LinQ CONFIRM_BOD BODs” on page 162.

For APIs using the OAGIS standard to exchange data, Q/LinQ generates a CONFIRM_BOD based on the value of the CONFIRMATION flag in the control area of the BOD conveying the data. The CONFIRM_BOD conveys the autoacknowledgment.

For APIs not using the OAGIS standard to exchange data, the Q/LinQ autoacknowledgment level determines when Q/LinQ generates or requests an autoacknowledgment message.

The autoacknowledgment level specifies which types of events generate the autoacknowledgment messages that Q/LinQ sends or receives. Level has three possible values:

All: Send a message whether the documents were successfully processed or failed to be processed.

Error: Send a message only if the processing failed.

None: Do not send any autoacknowledgment messages automatically.

For imported documents, any autoacknowledgment level set in the document prevails. If there is no level set in the document, the autoacknowledgment level set in Q/LinQ applies.

For exported documents, the autoacknowledgment level set in Q/LinQ applies.

- Use Q/LinQ Control (36.8.24) to establish the default autoacknowledgment level for all export and import documents.
- Use Register External Application (36.8.1.1; 36.8.1 in 9.0) to set the autoacknowledgment level for all documents exchanged with a particular external application. This setting defaults from Q/LinQ Control.
- Use Export Specification Maint (36.8.1.2; 36.8.2 in 9.0) or Import Specification Maint (36.8.1.3; 36.8.3 in 9.0) to set the acknowledgment level for particular export or import documents. These settings default from Register External Application.

See “Defining Control Program Values” on page 104 for details.

Note For export documents, the publishing API can override the level set in the control program or for the application or document.

Acknowledgment Processing

Q/LinQ processes the autoacknowledgment through user interface (UI) emulation using Process Received Acknowledgment (36.8.12). This procedure records the Q/LinQ receipt of the autoacknowledgment and logs it with the other records of the document.

Before Process Received Acknowledgment can function, you must write a CONFIRM_BOD and register it with Q/LinQ in Import Specification Maint (36.8.1.3; 36.8.3 in 9.0). Process Received Acknowledgment normally executes as part of Process Import Documents, and users do not need to invoke it.

E-Mail Notification

In addition to logging an autoacknowledgment, Q/LinQ can send e-mail to an MFG/PRO user when it successfully processes or fails to process an import document. The e-mail identifies the document using the Q/LinQ-generated document ID and describes the event that caused the e-mail. The messages are sent using the MFG/PRO e-mail interface.

The e-mail interface can connect to any e-mail system that can send messages from the UNIX or DOS command line. Define the settings and defaults in the same programs as those for defining autoacknowledgment messages:

- Q/LinQ Control
- Register External Application
- Export Specification Maint
- Import Specification Maint

Exchanging Documents

This section discusses two Q/LinQ methods for exchanging documents between MFG/PRO and an external application:

- Passing text files
- Using APIs

Document Exchange with Text Files

Passing data text files is simple and easy to implement. Although this method is the most commonly used, it does have drawbacks:

See Appendix B, “Text File Control Tags,” on page 167.

- The Q/LinQ-format data in exported files must be readable by the external application without an adapter.

- To communicate information about the nature of each document and to clearly separate the documents, the data in incoming files contain several Q/LinQ-specific document control tags before and after each document.
- Text files are often passed in batches. Batch processing of files, although it can be used with Q/LinQ, is not well suited to a primarily event-driven environment such as Q/LinQ.
- Passing files means more work for information systems and those who support them. It places an additional input or output load on the system and can generate many transient work files. These files increase housekeeping and directory-management duties.

Document Exchange with APIs

With the rise of object-oriented software architecture, standards, interoperability, and a need for more rapid, direct, event-driven communication between applications, APIs are increasingly the method of choice for data exchange.

Software vendors increasingly provide APIs to accommodate integration. Integration is also supported by various interoperability standards, which include:

- Extensible Markup Language (XML)
- Common Object Request Broker Architecture (CORBA)
- Open Application Group Integration Specification (OAGIS)
- Distributed Component Object Model (DCOM)

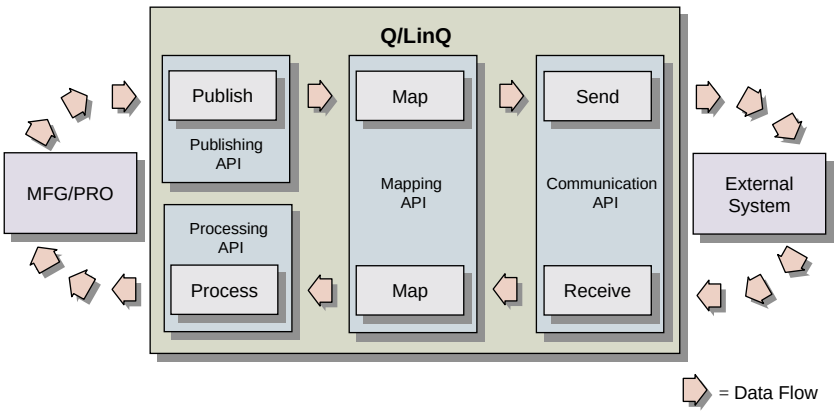
Combined with MOM, which provides message transport and remote procedure call facilities, standardized APIs can now support scalable, reliable, and performance-oriented distributed applications in heterogeneous environments.

The procedure callout (exit) points and callback (entry) points of APIs reduce the resources and time spent integrating applications.

Q/LinQ APIs

There are four types of Q/LinQ APIs: publishing, processing, mapping, and communication. Figure 10.3 shows the data flow through these APIs and the following sections describe each of the API types.

Fig. 10.3
Import and Export Data Flow Through Q/LinQ APIs



Note In all text and graphics, MFG/PRO and Q/LinQ are the point of reference. Imported documents are always documents that Q/LinQ receives from an external application for MFG/PRO. Exported documents are sent from MFG/PRO by way of Q/LinQ to an external application.

Publishing API

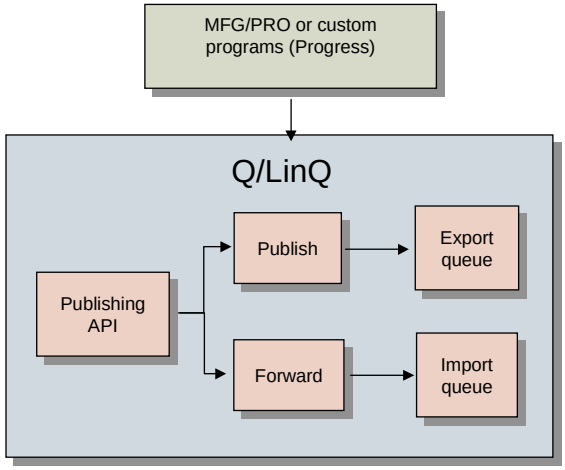
The Q/LinQ publishing API creates documents from MFG/PRO data for export to an external application or file. In addition, the API can export documents from one MFG/PRO module and place them in the Q/LinQ import queue for processing by another MFG/PRO module in the same MFG/PRO system. This operation is known as *forwarding*.

Forwarding lets MFG/PRO programs submit work for later processing outside their own transaction scope. This provides, in effect, an asynchronous transaction processing facility inside MFG/PRO.

Figure 10.4 illustrates how Q/LinQ publishes or forwards transaction data in a document. The shared routines for publishing and forwarding use a passed parameter to determine whether to prepare a document for publishing or forwarding. The routines then place the documents in the appropriate queue.

The publishing API replaces the write or export operations generally used to flow documents, or interface data, from MFG/PRO programs into a text file.

Fig. 10.4
Publishing API



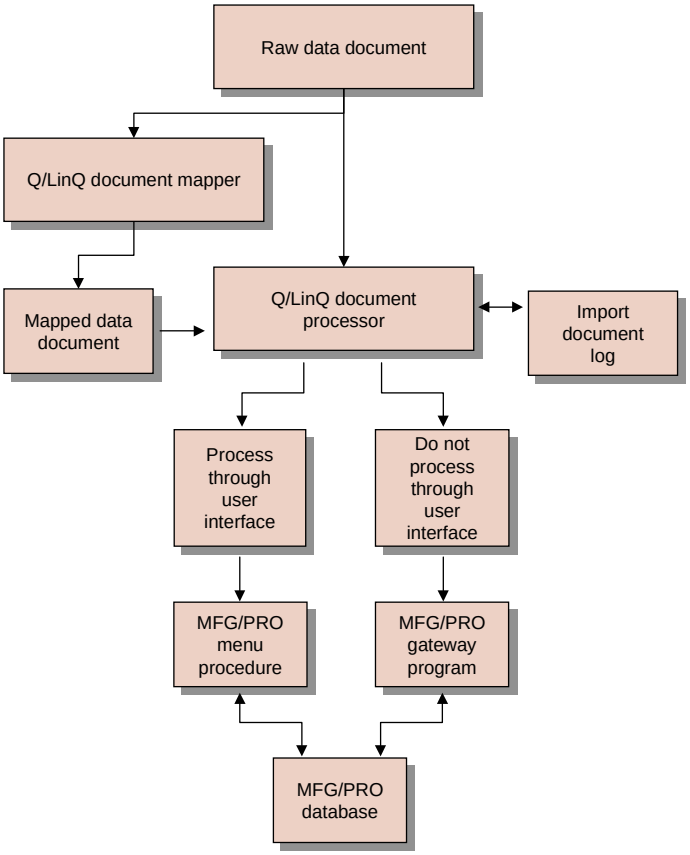
Processing API

The Q/LinQ processing API processes imported documents as MFG/PRO transactions. The documents must be imported in or mapped to a format recognized by MFG/PRO.

The processing API has two modes (see Figure 10.5):

- Emulate user input to MFG/PRO menu procedures (process through user interface).
- Directly load the database through gateway programs (do not process through user interface).

Fig. 10.5
Processing API



User Interface Emulation

User interface emulation uses MFG/PRO menu procedures to process documents from a file or an external Progress procedure.

The location of information on the CIM Interface is described on page 2.

This mode takes advantage of the standard MFG/PRO validation and update logic applied when the user interactively enters data. It traps error and warning messages. The Q/LinQ UI emulation mode is similar to the functions of the CIM Interface. The Q/LinQ interface processes data faster and has more options than the CIM interface, although with the UI emulation overhead, it still processes data more slowly than direct updates to the database.

When importing data from a file, the UI emulation mode can import existing customer-written CIM Interface and application data interface (ADI) programs.

UI emulation generally is not feasible or practical for complex documents, such as those processed by MFG/PRO programs with UI frames and pop-up windows that conditionally display based on application data. Neither is it suitable for importing large numbers of documents.

For example, UI emulation is not suitable for importing sales orders. The slower throughput, difficulty in trapping and handling errors, and brittle and error-prone data format of these types of transactions make direct database access through user-written gateway programs superior.

Direct Database Access

The direct database access mode uses gateway programs. These programs have no user interface; they directly load documents into the MFG/PRO database.

These user-written Progress procedures are analogous to the MFG/PRO Electronic Data Interchange (EDI) gateway programs. Gateway programs must duplicate the MFG/PRO validation and update logic of the procedure they bypass.

When processing a document, Q/LinQ invokes the program associated with that document type or the originating external application in Import Specification Maint (36.8.1.3; 36.8.3 in 9.0). Based on this association, Q/LinQ can call existing MFG/PRO EDI or EDI ECommerce gateway processing programs or user-written gateway processing programs.

As with any API to a separate executable process, the processing program must have been previously defined in Register External Application (36.8.1.1; 36.8.1 in 9.0).

Mapping API

See “Importing, Mapping, and Processing Data” on page 136 for the MFG/PRO format.

The mapping API transforms data from the format used in a source application to the format used in a target application.

Note When importing data to MFG/PRO, it is acceptable to use existing programs to transform, convert, or map the data into a format MFG/PRO can read. Invoke these processes before passing the imported data to Q/LinQ.

The mapping API (see Figure 10.6) invokes custom or QAD-supplied data-transformation programs that:

- Map published documents from MFG/PRO format into the external application format for export.
- Map imported documents from the external application format into MFG/PRO format.

Depending on the entry in Register External Application (36.8.1.1; 36.8.1 in 9.0), Export Specification Maint (36.8.1.2; 36.8.2 in 9.0), or Import Specification Maint (36.8.1.3; 36.8.3 in 9.0) that defines the default mapping procedure, the Q/LinQ mapping API calls:

- A user-written mapping routine written in Progress 4GL

See *Technical Reference: Q/LinQ*.

- A third-party mapping package, which requires its own custom adapter that a programmer designs and writes using the Q/LinQ mapping API

Third-party mapping tools are most appropriate for transforming more complex, self-describing data, such as OAGIS BODs or EDI documents. For most proprietary external-application data formats (flat, fixed-position), a Progress routine suffices.

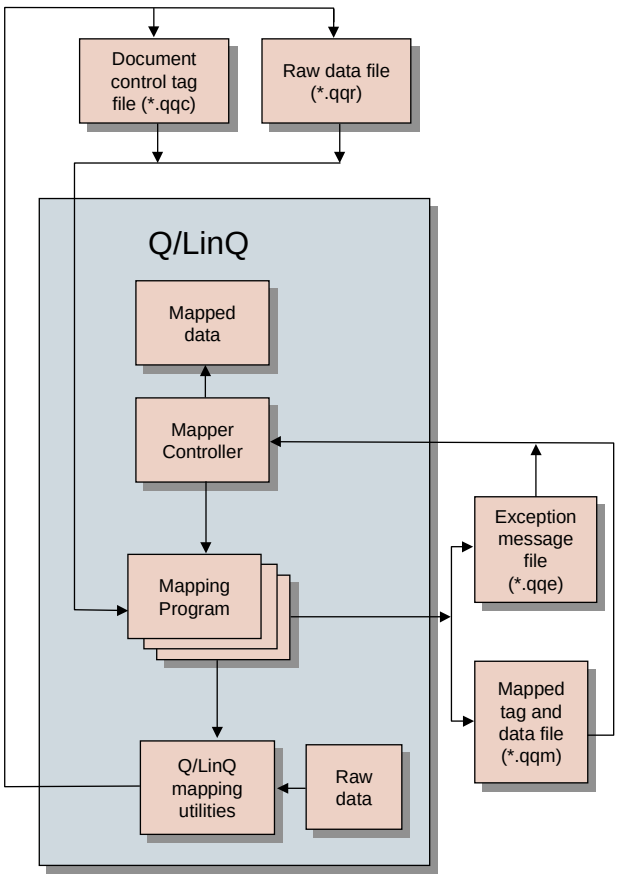


Fig. 10.6
Mapping API Architecture

For applications with complex data mapping, Q/LinQ supports XSLT stylesheets. For setup instructions, see:

- Chapter 3, “Communication and Mapping Setup on UNIX,” on page 21
- Chapter 5, “Communication and Mapping Setup on Windows,” on page 41

The external application receiving or sending the documents must be previously registered with Q/LinQ. Data mapping automatically occurs for import documents if the registry entry for the external application contains mapping definitions and the APIs exist. For export documents, the user can select to map and send the documents in one step or two.

See “Registering External Applications” on page 107.

Using Send Export Documents (36.8.7) or Process Import Documents (36.8.10) invokes the mapping tool.

Import Document Mapping

See “Importing, Mapping, and Processing Data” on page 136.

Initiate mapping in Process Import Documents (36.8.10). Select the documents to map by entering a range of document IDs, source applications, trading partners, revision levels, published standards, or document types.

One-to-One Mapping

Using the selection parameters, Q/LinQ claims a set of documents from the import queue. It selects all the documents that must be processed as a group (a batch or series) for the mapping session. It calls the data mapper routine, which transforms the documents. The raw import documents are always left intact.

When Q/LinQ encounters errors, it does not create mapped import document records. Finally, Q/LinQ updates the import log table to reflect the mapping activity, the time it occurred, the current status of the document, and messages generated by the processing.

One-to-Many Mapping

In some cases, a raw document requires the generation of multiple mapped documents for processing. For example, an OAGIS BOD might require two or more MFG/PRO maintenance procedures to fully process it. The mapping routine generates the required number of mapped documents, one for each MFG/PRO function.

In one-to-many mapping, the mapping process creates multiple mapped import document records that share the same document ID and header information. Q/LinQ uses a document suffix field to differentiate among the mapped documents.

See “Transaction Document Groups” on page 84.

To preserve document integrity, Q/LinQ redefines a document group if it is mapped to multiple import documents. For example, for a raw document imported as an individual-document group and then mapped into three documents, Q/LinQ defines the three mapped documents as members of one batch group.

Export Document Mapping

For Q/LinQ, mapping export documents is less complex than mapping import documents. The external application performs the mapping that Q/LinQ did for import documents. When users select Map Raw Documents in Send Export Documents (36.8.7), Q/LinQ calls the registered mapping API, maps the data, and sends it to the external application, which takes any further steps necessary to transform the data.

Communication APIs

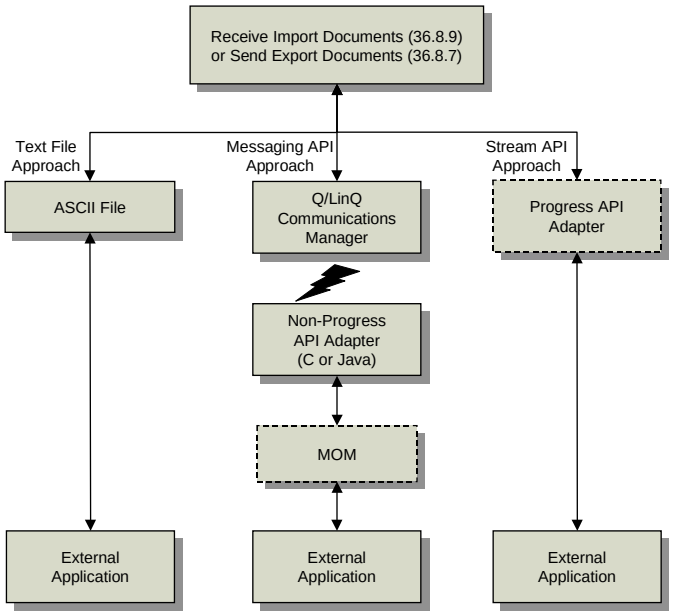
The communication APIs transfer documents directly to or from external applications, rather than dumping or loading documents to or from text files.

There are two types of communication APIs, messaging and stream:

- Use the messaging API—C and Java routines—for interprocess communications with MOM products or with non-Progress applications. Use MOM to ensure secure communications when exchanging data outside the company network. The messaging API uses an adapter—based on TCP/IP Sockets and Windows Sockets (Winsock)—to communicate with the middleware.
- Use the stream API—Progress routines—for writing to or reading from files that do not conform to the Q/LinQ format, for calling a non-MFG/PRO Progress routine to transfer the data, or for calling Q/LinQ from a non-MFG/PRO Progress program.

Figure 10.7 illustrates the communication process.

Fig. 10.7
Importing from or Exporting to a Process or Text File



The communication APIs support only loosely coupled, asynchronous communications. They do not immediately send responses or return codes related to the results of MFG/PRO validation or transaction processing.

The APIs support interface connections that can concurrently send to and receive from the same external application. However, this two-way traffic is not synchronized and must occur within separate MFG/PRO log-in sessions.

Q/LinQ supports multiple concurrent export and import activities with multiple external applications. However, only one Q/LinQ export or import session can export to or import from a given external application at a time. For example, one Q/LinQ session can export to one application while another Q/LinQ session simultaneously receives imports from the same application, but the two Q/LinQ sessions cannot import from the same application at the same time. This limitation preserves the document sequence for processing by the external application.

Note Each concurrent Q/LinQ session runs from a separate MFG/PRO log in and thus increases the user count required on the MFG/PRO license.

Message-Oriented Middleware

The messaging communication API can use middleware to communicate with the external application. Middleware is a general term for distributed computing and application development environments. It enables distributed applications across multiple machines, operating systems, and networks.

Message-oriented middleware (MOM) is a particular type of middleware that provides secure communication and assured message delivery among the distributed application processes. Q/LinQ supports MQSeries, the MOM from IBM. The messaging communication API uses an adapter to communicate with the MOM.

MOM Adapter

The Q/LinQ MOM adapter, `QqMomAdapter`, is a program written in Java that mediates communication between one or more MQSeries queues and one or more Q/LinQ sessions. It can reside on the Q/LinQ machine, a MOM machine, or a third machine.

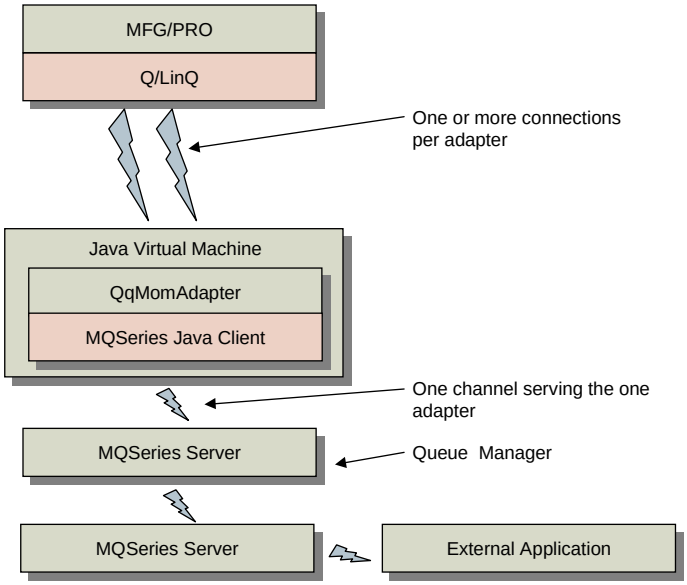
Note Running the MOM adapter on the same machine as Q/LinQ is recommended because it simplifies system administration.

The adapter is a TCP/IP listener that can handle one or more client connections in a multi-threaded manner.

Although using the Java programming language gives the adapter platform independence, the MOM can impose some platform limitations. Consult the MQSeries Java client documentation to determine any constraints.

Example Figure 10.8 shows a distributed MQSeries configuration with separate servers in remote locations. One MQSeries server is on the Q/LinQ side and contains the queue manager; one or more MQSeries servers are at other locations and connected to external applications.

Fig. 10.8
Using MQSeries with the Messaging API



Chapter 11

Setting Up Q/LinQ

This chapter is for the system administrators who set up Q/LinQ.

Overview **104**

Reviewing the Initialization File **104**

Defining Control Program Values **104**

Registering External Applications **107**

Setting Up Document Specifications **115**

Defining Code Mappings **116**

Starting and Restarting Q/LinQ **118**

Starting and Stopping the MOM Adapter **119**

Testing Communication **122**

Overview

Q/LinQ requires setup by the MFG/PRO system administrator to enable data exchange with external applications.

After installation, set parameters in the Q/LinQ Control. This is the only setup task required if using Q/LinQ for only text-file dump/load or for UI-emulation.

Exchanging data directly with external applications requires the following additional tasks:

- Register each external application with Q/LinQ and enter data-mapping parameters used to transform data.
- Enter specifications for export and import documents.

After registration, complete the Q/LinQ setup with these tasks:

- Start the message-oriented middleware adapter, if used.
- Test communication between Q/LinQ and the external application.

Reviewing the Initialization File

See Chapter 7.

Some values that MFG/PRO uses to configure Q/LinQ are specified during installation. These values are stored in an initialization (`.ini`) file. They include the location where Q/LinQ saves log, archive, and audit files, and the location of programs that Q/LinQ uses. You may need to review or update these values to ensure that they are appropriate.

Defining Control Program Values

Before you can exchange data using Q/LinQ, you must specify several values in Q/LinQ Control (36.8.24). The two frames on the first screen maintain system parameters and e-mail communications defaults.

After setting control parameters, log out of MFG/PRO and log on again before attempting to use Q/LinQ to receive, process, or send documents.

Note Although the figures in this chapter show a particular version number, the instructions apply to all versions noted on the title page.

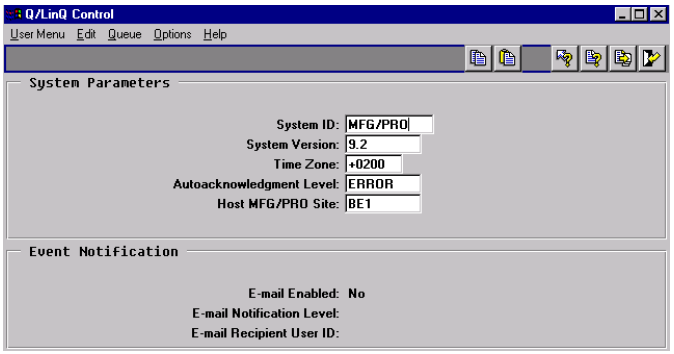


Fig. 11.1
Q/LinQ Control (36.8.24), System Parameters and Event Notification

System ID. Enter the logical ID of this MFG/PRO installation. This is a static, global value that identifies the source MFG/PRO system on exported documents. It is also entered as the destination application ID when forwarding documents from MFG/PRO procedures to other MFG/PRO procedures using the publishing API.

System Version. Optional. Briefly identify the version or release level of the source MFG/PRO system. This description appears on exported documents (maximum 8 characters).

Time Zone. Enter the time zone associated with this MFG/PRO database server (maximum 5 characters). Use the generalized-code values in the lookup. This time zone is the basis for the date-time stamp in the import and export log file.

Autoacknowledgment Level. Enter the type of automatic processing confirmation for Q/LinQ to send to or receive from the external application.

See page 87.

- All: Sends an acknowledgment whether the import or export succeeded or failed.
- Error: Sends an acknowledgment if it failed.
- None: Sends no acknowledgment.

Autoacknowledgments include a date and time stamp. Q/LinQ saves them in the import or export log of the document.

The OAGIS CONFIRM_BOD BOD is a primary example of an autoacknowledgment document.

See “Q/LinQ CONFIRM_BOD BODs” on page 162.

The value specified here defaults to Register External Application, which defaults to Export Specification Maint and Import Specification Maint.

Host MFG/PRO Site. Optional. Briefly describe the primary MFG/PRO host site this Q/LinQ installation is associated with (maximum 8 characters). Use when Q/LinQ is deployed with multisite MFG/PRO implementations and the site code on MFG/PRO documents is populated. This field is not validated against Site Maintenance (1.1.13).

E-mail Enabled. Enter Yes to send an e-mail notification of Q/LinQ events to a user. This value applies to the entire system, not just one session or one application. Designate the user to receive messages in E-mail Recipient User ID. Specify the event that causes e-mail in E-mail Notification Level.

E-mail Notification Level. Q/LinQ sends e-mail for the type of processing event you choose:

- None: No e-mail for any event
- Error: E-mail only when Q/LinQ encounters errors during processing
- Warning: E-mail when Q/LinQ encounters either errors or warnings
- All: E-mail for errors, warnings, and successful processing

This value defaults to Register External Application, which defaults to Export Specification Maintenance and Import Specification Maintenance.

E-mail Recipient User ID. Enter the MFG/PRO user ID of the person to notify by e-mail of successful processing of Q/LinQ documents or of errors or warnings encountered during processing of the documents (maximum 8 characters). The ID must have been previously defined with User Maintenance (36.3.18).

The Messaging Defaults frame sets optional message-oriented middleware (MOM) parameters that are not currently implemented.

Registering External Applications

External applications must be registered in Q/LinQ (36.8.1.1, 36.8.1 in 9.0) in order for Q/LinQ to do any of these activities:

- Publish data from MFG/PRO to create export documents for the external application.
- Export documents to the external application.
- Map data imported from or exported to the application.
- Communicate directly with the application or message-oriented middleware (MOM).
- Dynamically determine which MFG/PRO program processes the imported data.

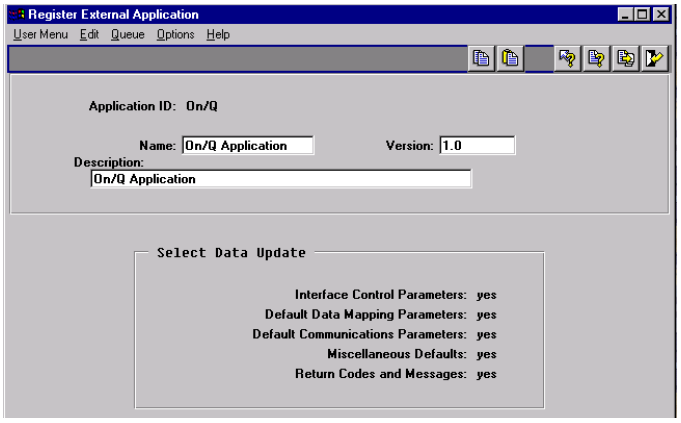
Registering an application requires the following:

- Entering the type of interface activity between Q/LinQ and the external application; for example, publish and send exports but do not receive, map, or process imports
- Setting mapping and communication parameters
- Defining the transaction processing program
- Mapping the return codes and messages of the external application, if any, to MFG/PRO messages

Identify Application

In Application ID, enter a unique ID for the external application or MOM. Enter a name and a brief description. In the Select Data Update frame of the main screen, select which frames have data that you want to update and press Go. The frame for your first selection appears. Make your edits, and press Go. The next frame displays, and so on. Press End at any time to return to the main screen.

Fig. 11.2
Register External Application (36.8.1.1, 36.8.1 in 9.0), First Screen



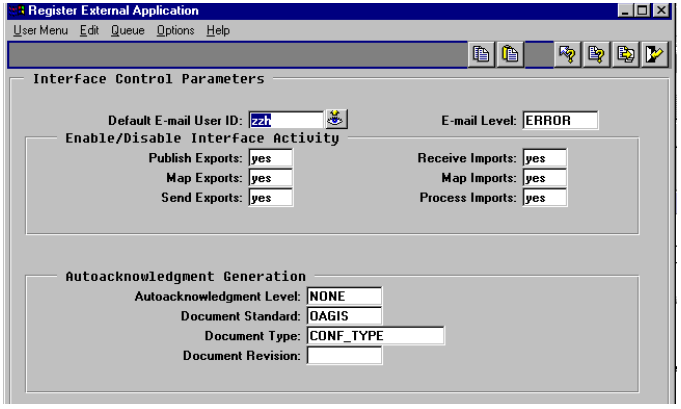
Interface Control Parameters

In the Interface Control Parameters frame, set the e-mail notification level and specify the MFG/PRO user to receive messages. Specify whether to publish, receive, map, send, or process the documents. Enter the level of event that induces autoacknowledgment of document processing and the standard the autoacknowledgment uses; for example, OAGIS.

See page 104.

The Default E-mail User ID, E-mail Level, and Autoacknowledgment Level default from values previously defined in Q/LinQ Control.

Fig. 11.3
Register External Application, Interface Control Parameters



Data Mapping Parameters

In the Default Data Mapping Parameters frame, define the data-mapping program, the particular map, and the map parameters and values. If you are using XSLT stylesheets, enter specific values in Export Specification Maint (36.8.1.2; 36.8.2 in 9.0) and Import Specification Maint (36.8.1.3; 36.8.3 in 9.0).

Define the ASCII delimiter for the triplet format used during import and export. The triplet delimiter must be an ASCII character that does not occur in the document data. Q/LinQ inserts the triplet delimiter between data triplets when generating a triplet-formatted document for export and uses the delimiter to parse an imported document into data triplets.

Note This feature is not available in MFG/PRO 9.0.

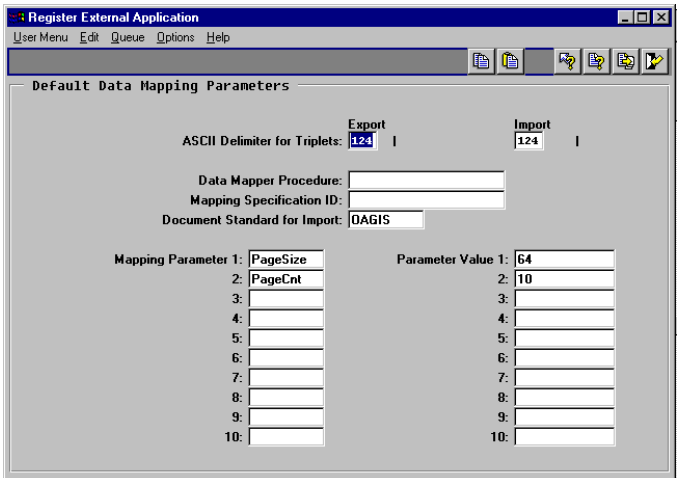


Fig. 11.4 Register External Application, Default Data Mapping Parameters (non-9.0)

Communication Parameters

In the Default Communications Parameters frame, the Access Code/Path, such as an authentication ID or name, is any value required by an API to open a session with the registered application. This could be a user ID and password pair or security-related key, although it is not encrypted. The Access Code/Path could also be a directory name or URL that is meaningful to your communications adapter program.

Q/LinQ communicates with the external applications through custom interface adapters—specifically written for the external application or system—that use direct procedure calls or interprocess communication.

If the external application is a Progress program, enter the custom adapter to use in the Stream API Adapter Program field.

If communicating with non-Progress programs, enter the messaging API socket parameters. You do not need to enter a program name for a messaging API; specifying its parameters is sufficient. These can also be values for communicating with the MOM.

See “Trading Partners and MOM Channels or Queues” on page 116.

Each adapter using the messaging API requires a TCP/IP port for exporting and another for importing. However, multiple applications can share the same port if the adapter can route the documents between Q/LinQ and the various source or destination systems. For example, a MOM can communicate with several external applications if each application has a unique trading partner ID.

The Quality of Service (QOS) options are intended for use with MOM products. Quality of service options can vary greatly among commercial MOM products. Override the values in Q/LinQ by application in Register External Application or by document in Export Specification Maint or Import Specification Maint.

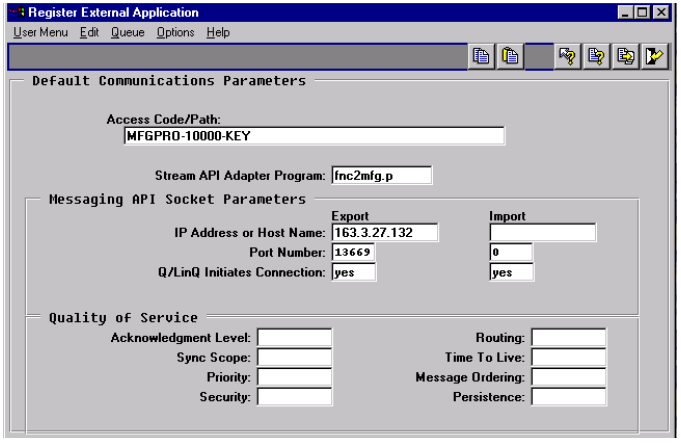


Fig. 11.5
Register External Application, Default Communications Parameters

Miscellaneous Defaults

In the Transaction Processing Programs frame within the Miscellaneous Defaults frame, enter the program to which the adapter sends the documents for processing.

- For the field In Registered Application, if the external application API or adapter program requires it, enter the name of the program in the external application that is to process export documents. Otherwise, this is optional.
- For the field In MFG/PRO, enter the program in MFG/PRO that is to process import documents.

If the transaction data are to be processed by UI emulation, enter Yes in Process Through User Interface.

Leave the default No for Process Outside of Transaction if you want Q/LinQ to manage all imported documents as one transaction. When Process Outside of Transaction is Yes, the failure of one document does not affect the entire import transaction. Setting this field to Yes may provide you with better performance since the system does not need to log the information required to roll back multiple uncommitted changes. This field sets the default for a similar field in Import Specification Maintenance.

The other fields in the frame are not implemented or are for optional reference information.

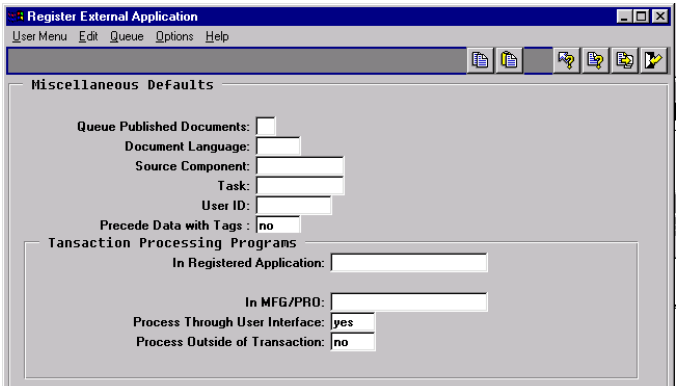


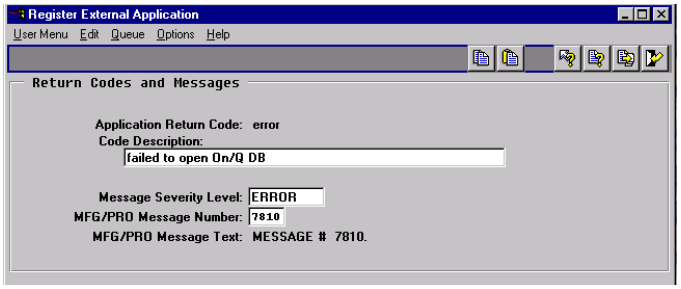
Fig. 11.6
Register External Application, Miscellaneous Defaults

Return Codes and Messages

In the Return Codes and Messages frame, map external application return codes to MFG/PRO messages. These return codes must be the same ones returned by the adapter program to the Q/LinQ messaging or stream API.

Based on mappings in this frame, external adapters can return exception codes to Q/LinQ that are not the same as any MFG/PRO message numbers, but that translate into predefined MFG/PRO or Q/LinQ messages.

Fig. 11.7
Register External Application, Return Codes and Messages



Registering Multiple Applications

There are different approaches for registering multiple applications depending on whether the communication API uses MOM.

Registering Multiple Applications Without MOM

Register a system of multiple external applications as a single application. For example, register another MFG/PRO installation as a single application.

Treat each installation of the same application at different sites as a distinct application. For example, treat each site that runs the same warehousing software as a separate application.

Registering Multiple Applications with MOM

With MOM there are two approaches to configuring communication between Q/LinQ and multiple applications. In this example, the MOM is MQSeries.

First Example

A single Q/LinQ external application ID corresponding to MQSeries is used to send and receive messages through MQSeries with all external applications, as illustrated in Figure 11.8.

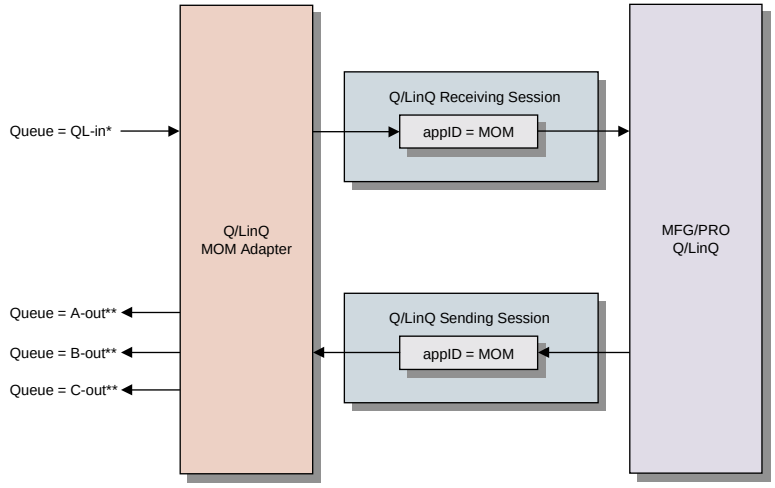


Fig. 11.8
Single External Application ID

Messages coming into Q/LinQ are received from one queue and identified by the value of the trading partner ID. Messages sent to MQSeries are routed to the appropriate queue based on the trading partner ID value of the message. If no trading partner ID value is present in the message, the value of the external application ID is used to determine the destination queue. The `QqMomAdapter.ini` file provides this mapping.

See “Q/LinQ MOM Adapter Initialization File” on page 61.

To set up this kind of system, follow these steps:

1. Register the MOM as the external application in Register External Application.
2. In the Default Communications Parameters frame, assign the MOM to a single port number, indicating the port where the MOM adapter is running.

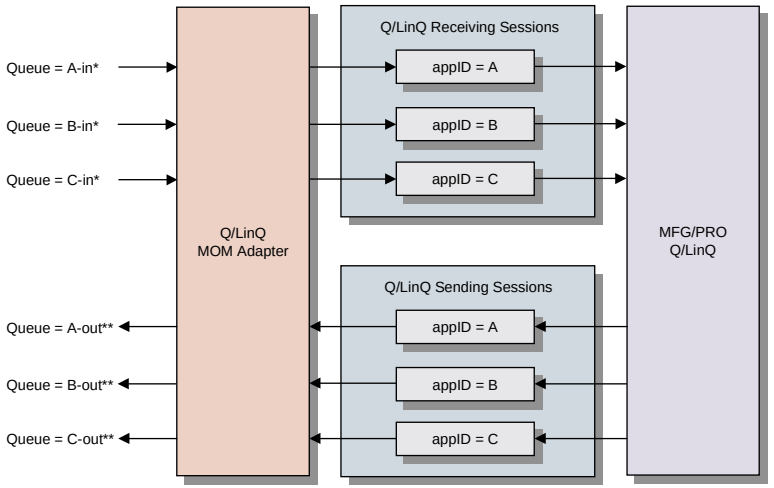
3 In Export Specification Maint and Import Specification Maint, create a record for each application with the following values:

- Set application ID to the registered MOM.
- Set trading partner ID to each individual application.

Second Example

In the example in Figure 11.9, a separate Q/LinQ external application is used to exchange messages with each external application.

Fig. 11.9 Separate External Application IDs



* Inbound: Queue is mapped to appID
** Outbound: Trading Partner ID (tradPtrID) or Application ID (appID) is mapped to queue

See “Q/LinQ MOM Adapter Initialization File” on page 61.

Each external application has a queue that is mapped one-to-one with a Q/LinQ external application ID. The `QqMomAdapter.ini` file provides this mapping.

To set up this kind of system, follow these steps:

1 Register each application separately in Register External Applications.

2 In the Default Communications Parameters frame, assign each application the same port number. This is the port where the MOM adapter is running.

Setting Up Document Specifications

Use Export Specification Maint (36.8.1.2; 36.8.2 in 9.0) and Import Specification Maint (36.8.1.3; 36.8.3 in 9.0) to specify values used when specific documents are imported or exported. These values include data mapping values, miscellaneous defaults, the source of the transaction data, and the procedure to process the data.

If there are no Import Specification Maint or Export Specification Maint records for a document, Q/LinQ refers to Register External Application for any required parameter values.

Document Identification

Document identification includes document standard, type, revision, and trading partner.

See “Trading Partners and MOM Channels or Queues” on page 116 regarding special use for the Trading Partner ID field.

A document standard defines a data architecture, or format, that describes the structure and content of the document. Examples of standards are MFG/PRO CIM, ANSI X12, EDIFACT, OLE/COM, and OAGIS. Some standards contain specific document types. For example, the OAGIS standard has CONFIRM_BOD and UPDATE_INVENTORY document types.

The revision level of the document is an OAGIS-specific entry. It denotes the Business Object Document (BOD) version. It increases each time the BOD specification changes.

If the external application complies with OAGIS, you can define OAGIS as the default document standard for import documents. In this case, the external application does not need to provide document tags. Q/LinQ can parse imported BODs to obtain the document type and revision.

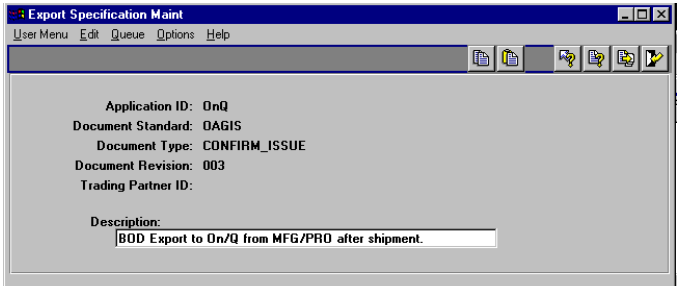


Fig. 11.10 Export Specification Maint (36.8.1.2, 36.8.2 in 9.0), First Screen

In the subsequent frames of Export Specification Maint, specify the interface control, data mapping, and messaging parameter values that distinguish this specific document from other documents used by the same external application. Set up similar parameters in Import Specification Maint for specific imported documents.

Trading Partners and MOM Channels or Queues

See "Registering Multiple Applications with MOM" on page 112.

When multiple applications are using a single MOM channel or queue to communicate with Q/LinQ, use the Trading Partner ID in Export Specification Maint (36.8.1.2; 36.8.2 in 9.0) and Import Specification Maint (36.8.1.3; 36.8.3 in 9.0) to differentiate among messages from the various applications.

Defining Code Mappings

Note Code mapping is available for all versions of MFG/PRO except MFG/PRO 9.0.

For some variables or codes, the values used in external applications are different from those used in MFG/PRO. In these cases, the values must be translated or mapped when exchanged between the applications.

Use Code Mapping Maintenance (36.8.1.20) to define the mapping for specific code values between MFG/PRO and external applications. All values are expressed as ASCII character strings; no other character types are supported. The mapped code values are stored in a translation table.

See *External Interface Guide: Logistics*.

In Logistics, mapping is supported for the CHARGEID element in BODs and trailer charges in MFG/PRO.

Software developers writing programs using the Q/LinQ publishing, mapping, or processing APIs, can use the API function `qqMappedCode` to obtain the translated value of an input value for any code maintained in Code Mapping Maintenance.

The fields Application ID, Document Standard, Document Type, Document Revision, and Trading Partner ID are optional. Together, the values in these fields form a search key to identify the set of documents to which the mapping applies.

At runtime, when importing or exporting BODs, MFG/PRO searches the translation table for a mapped code value, using the search keys applicable to the current document. It uses the most specific search key first. If no value is returned, it continues searching using progressively less specific search keys and returns the first value found.

If MFG/PRO finds no match in the translation table for the value being processed, it attempts to use the untranslated value. When importing BODs, if the value that MFG/PRO attempts to use is not an expected value, there is an error and the BOD is not processed.

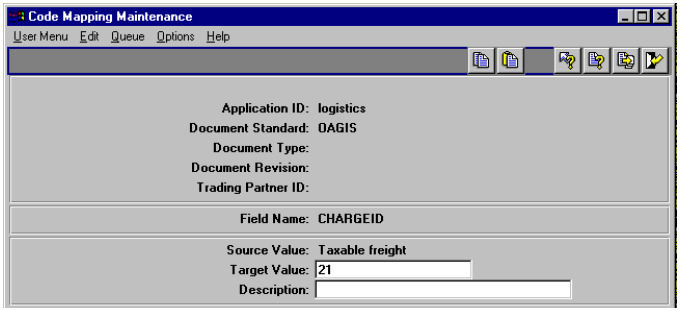


Fig. 11.11 Code Mapping Maintenance (36.8.1.20)

Application ID. Enter the name of an external application as it is defined in Register External Applications.

Document Standard. Optional. Enter a published standard (maximum 8 characters); the code mapping applies to documents using this standard. Enter CIM if the documents are formatted for stream input into a Progress program. The standard describes the structure and content of the document. Examples of standards are EDIFACT, OAGIS, and ANSI X12.

This field is optional if the document does not follow a published, revision-controlled standard such as OAGIS. This might be true of documents processed by custom legacy applications. Sometimes, document type is all that is required.

Document Type. Optional. Enter a document type; the code mapping applies to documents of this type. Examples include the 850 transaction from the ANSI X12 standard or the OAGIS verb and noun of the BOD, separated by an underscore; for example, UPDATE_INVENTORY.

Document Revision. Optional. Enter the revision number of the document type (maximum 8 characters); the code mapping applies to documents where this is the revision number for the document type. For example, in the case of X12 EDI, this is the revision number of a specific X12 document definition. In the case of an OAGIS BOD, it is the three-digit number denoting the revision level of the BOD.

Trading Partner ID. Optional. Enter an ID for the trading partner whose documents this code mapping applies to. This ID is not checked against any table of defined trading partners.

Field Name. For Logistics, the supported field is CHARGEID.

Source Value. Enter the code before mapping. All code values must be ASCII character strings; no other types are supported.

Target Value. Enter the code value after mapping. As with source values, all code values must be ASCII character strings.

Description. Enter the description of the mapping, for example, the rationale or the mathematical computation used.

Starting and Restarting Q/LinQ

Normal Start and Restart

Starting MFG/PRO invokes Q/LinQ. However, you must start and stop any communication adapters separately from MFG/PRO or Q/LinQ.

Restarting Q/LinQ after a normal system shutdown does not require any special steps: start MFG/PRO to invoke Q/LinQ.

Restart After System Failure

If a system failure, such as a power outage or a user interrupt, aborts a Q/LinQ session, follow these steps:

Running multiple sessions is discussed on page 100.

1. Open Interface Session Monitor (36.8.5) and cancel all sessions that were running at the time of the shutdown. A Q/LinQ session is a single Q/LinQ executing menu procedure that publishes, sends, receives, processes, archives, or deletes documents.

2 Stop and restart all communication adapters that were communicating with the interrupted session; for example, the adapters communicating with Receive Import Documents (36.8.9) or with Send Export Documents (36.8.7).

See “Starting and Stopping the MOM Adapter” on page 119.

3 Restart the Q/LinQ session.

Starting and Stopping the MOM Adapter

After the adapter has been started, it runs continuously, regardless of whether Q/LinQ is running, and does not need to be shut down before you exit from MFG/PRO. When an MFG/PRO session ends by normal means, Q/LinQ disconnects from the adapter. The adapter goes into listen mode, waiting for Q/LinQ to connect again.

Tip
Q/LinQ always initiates the connection with the adapter.

Starting the MOM Adapter

To use the MOM adapter, follow these steps.

1 Configure `QqMomAdapter.ini` if it was not configured during installation.

2 Start MFG/PRO.

3 Start the MQSeries server.

4 Start the listener process for the MQSeries channel created during installation. For example, in Windows:

See the MQSeries installation documentation.

```
Qlinq.Channel runmqlsr -t tcp -m QADQM -p 1414
```

Where:

- `-t` Transmission protocol
- `-m` Queue manager name
- `-p` Port to listen on

5 Launch the `QqMomAdapter`:

```
java QqMomAdapter [-i <iniFileName>]
```

If the optional `<iniFileName>` parameter is not specified, a default value of `QqMomAdapter.ini` is used.

Alternatively, for MOM adapters running on UNIX, start `QqMomAdapter` with the user-written shell script `qqLaunchAdapter.sh` (see Figure 11.12). Use the script rather than the `ps` command, which truncates the end of the command line and cuts off the port number. The script launches `QqMomAdapter` as a background process and writes its process ID to the log file, where you can obtain it.

Fig. 11.12 Script for Starting Adapters

```
#!/bin/sh
#
#this user-written shell script, qqLaunchAdapter.sh, launches the
#QqMomAdapter process in the background, and writes the
#QqMomAdapter Unix process ID into the log file.
QQLOGFILE=QqMomAdapter.log
java QqMomAdapter &
sleep 1
echo "*****************************************************" >>
${QQLOGFILE}
echo "Unix process ID: $!" >> ${QQLOGFILE}
echo "*****************************************************" >>
${QQLOGFILE}
echo "QqMomAdapter process launched. Process ID= $!"
exit 0
```

Stopping the MOM Adapter

Normally, `QqMomAdapter` runs continuously as a background process. If you must stop an adapter, such as after a system failure, use the following commands to stop and restart the adapter manually outside of Q/LinQ.

UNIX

If only one adapter is running:

1. Obtain the process ID:

   ```
   ps -ef | grep QqMomAdapter
   ```

2. Stop the process:

   ```
   kill <process ID>
   ```

If multiple adapters are running:

1. Obtain the process ID from the log files.
2. Stop the process:

   ```
   kill <process ID>
   ```

Windows

1 Select Close from the system menu.

2 Select End Task.

An alternative method is to press Ctrl+C.

Preserving Data if a Connection Breaks

To ensure that you do not lose data if the connection breaks between Q/LinQ and MQSeries while MQSeries is transmitting data, use the MQSeries SYNCPOINT option. The SYNCPOINT option ensures that data not yet received is rolled back into the MQSeries queue, and MQSeries knows it has not been sent.

The following code fragment from the MOM adapter illustrates how the adapter uses the MQSeries SYNCPOINT option with the MQGetMessageOptions command when making GET calls:

```
//
    // set message options:
    // - block
    // - set syncpoint for committing or rolling back the GET
    // - only block for "timeout" milliseconds. (infinite if
    // - timeout=0)
    //
    gmo = new MQGetMessageOptions();
    gmo.options = MQC.MQGMO_WAIT | MQC.MQGMO_SYNCPOINT;
    if (timeout == 0)
        timeout = MQC.MQWI_UNLIMITED;  // infinite blocking time
    gmo.waitInterval = timeout;      // timeout for 'get' (ms)
```

The default MQPutMessageOptions setting of MQPMO_DEFAULT is used when sending documents to MQSeries using PUT calls. The setting tells MQSeries that the persistence of a message inside the queue is determined by the persistence setting of the queue manager. Therefore, message persistence can be modified by changing the queue manager setting without reconfiguring the MOM adapter.

Testing Communication

After Q/LinQ is set up, test communication between Q/LinQ and non-Progress applications using the ANSI C (`qqsend`, `qqrecv`) or Java test programs (`QqSend.class`, `QqRecv.class`).

Note The C programs are in *`QlinqInstallDir`*`/resource/bin.` The Java programs are in *`QlinqInstallDir`*`/resource/classes`.

See “Q/LinQ MOM Adapter Initialization File” on page 61.

These test programs use the messaging API and MOM adapter to send prepared documents to Q/LinQ and to receive documents published and sent by Q/LinQ. Use them to diagnose communication problems, verify document integrity, and ensure that the behavior of the external application has been correctly emulated.

These programs can also be used to test custom adapters written to communicate with Q/LinQ via the messaging communication API in environments where Q/LinQ, MFG/PRO, or Progress are not installed.

See “Editing Document Control Tags” on page 148.

The send programs read a Q/LinQ control-tag file, process the tags, and send the data to a Q/LinQ receiving process. This receiving process can be:

- Receive Import Documents (36.8.9)
- `qqrecv`
- Another receive adapter. For Java, it could be MOM, using a `QqMomAdapter` export process.

The receive programs receive data from a Q/LinQ sending process and process the tags. This sending process can be:

- Send Export Documents (36.8.7)
- `qqsend`
- Another sending adapter. For Java, it could be MOM, using a `QqMomAdapter` import process.

Using the C Test Programs

Run `qqsend` and `qqrecv` from either a UNIX prompt or a Windows command prompt.

Running qqsend

Send test documents to Receive Import Documents (36.8.9) with `qqsend` in active or passive mode.

To run `qqsend`, you need the following:

- The IP address or host name of the server that Q/LinQ runs on
- The port number that Q/LinQ uses
- The external application ID of the Q/LinQ receiving session
- A Q/LinQ-formatted file

When using `qqsend` in active mode, Receive Import Documents must be running, set up to receive, and waiting for `qqsend` to initiate the connection. Use the following command:

```
qqsend 1 <host> <portNum> <appID> <datafile>
```

When using `qqsend` in passive mode, Receive Import Documents can initiate the connection. To use the passive mode, start the send program before the Q/LinQ process; then, use the following command:

See “Receiving Import Documents” on page 139 for details.

```
qqsend 0 <portNum> <appID> <datafile>
```

Running qqrecv

Receive test documents from Send Export Documents (36.8.7) using `qqrecv` in active or passive mode.

See “Forwarding to MFG/PRO” on page 131 for details.

To run `qqrecv`, you need the following:

- The IP address or host name of the server that Q/LinQ runs on
- The port number that Q/LinQ uses
- The external application ID of the Q/LinQ receiving session

When using `qqrecv` in active mode, Q/LinQ can wait for `qqrecv` to initiate the connection:

```
qqrecv 1 <host> <portNum> <appID>
```

When using `qqrecv` in passive mode, Q/LinQ can initiate the connection:

```
qqrecv 0 <portNum> <appID>
```

Using the Java Test Programs

Run `QqSend.class` and `QqRecv.class` from either a UNIX prompt or a Windows command prompt.

These programs are exactly the same as the C programs except for the command-line syntax.

When running `QqSend.class` in active mode, use the following command:

```
java QqSend 1 <host> <portNum> <appID> <datafile>
```

When running `QqSend.class` in passive mode, use the following command:

```
java Qqsend 0 <portNum> <appID> <datafile>
```

When running `QqRecv.class` in active mode, use the following command:

```
java QqRecv 1 <host> <portNum> <appID>
```

When running `QqRecv.class` in passive mode, use the following command:

```
java QqRecv 0 <portNum> <appID>
```

Chapter 12

Managing Documents

This chapter is for the MFG/PRO users who initiate or monitor data exchange between MFG/PRO and other applications. It provides the basic steps for exporting or importing data.

Processing Stage and Error Status **126**

Document Groups and Error Recovery **128**

Exporting Data **129**

Importing, Mapping, and Processing Data **136**

Monitoring Q/LinQ Sessions **145**

Editing Documents, Data, Control Tags **146**

Dumping Data to a File **150**

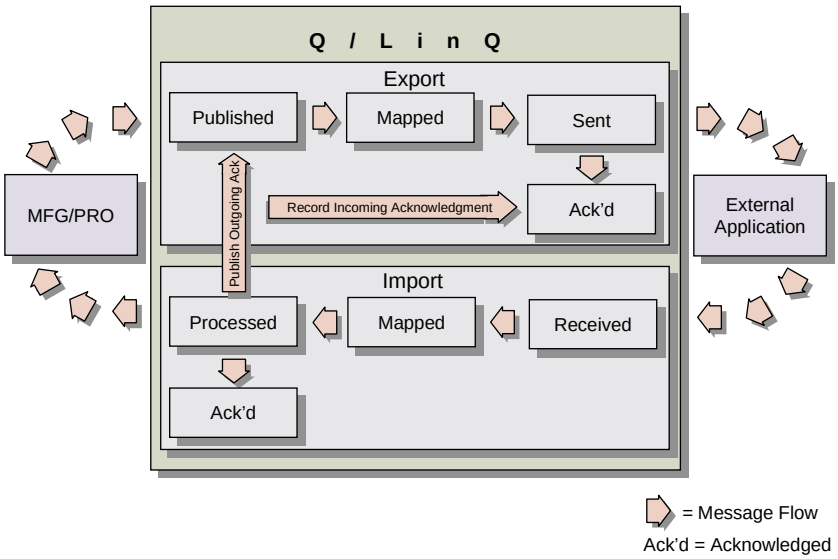
Generating Reports **151**

Deleting or Archiving Documents **153**

Processing Stage and Error Status

Figure 12.1 shows the various processing stages that documents pass through during export and import. Each document also has a current error status indicating whether it has successfully moved into its current processing stage.

Fig. 12.1
Export/Import Processing Stages



Assigning Processing Stages

Q/LinQ assigns documents to processing stages based on where they are in the export or import process flow. For all MFG/PRO versions, Q/LinQ assigns the *processed* and *sent* processing stages in the same way. For the other processing stages, there are differences between MFG/PRO versions in how Q/LinQ assigns documents to them.

Table 12.1 shows the version differences in assigning documents to export processing stages. For the entire export process, see Figure 12.3, “Export Process Flow for 9.0,” on page 129 and Figure 12.4, “Export Process Flow for MFG/PRO eB and eB2,” on page 130.

Table 12.1 Export Processing Stages

Export Stage	MFG/PRO eB and eB2	MFG/PRO 9.0
Published	Documents that have not been mapped or sent	Documents that are not registered to be mapped
Mapped	Documents that have been mapped	Documents that are registered to be mapped whether they have been mapped or not

Table 12.2 shows the version differences in assigning documents to import processing stages. For the entire import process, see Figure 12.7, “Import Process Flow for 9.0,” on page 137 and Figure 12.8, “Import Process Flow for MFG/PRO eB and eB2,” on page 138.

Table 12.2 Import Processing Stages

Import Stage	MFG/PRO eB and eB2	MFG/PRO 9.0
Received	All received documents	Documents that are registered to be mapped
Mapped	Documents that have been mapped	Documents that have been mapped or are not registered to be mapped

Note In MFG/PRO 9.0, a document can be mapped only if it is registered to be mapped in Register External Application (36.8.1) or Import Specification Maint (36.8.3) *before* it is imported. If the registration is missing, complete it and re-import the document.

Reviewing Document Processing Stage and Error Status

Use Export/Import Document Query (36.8.16) to review processing stage and error status. Taken together, processing stage and error status specify the last stage that a document was assigned to and whether the action was successful.

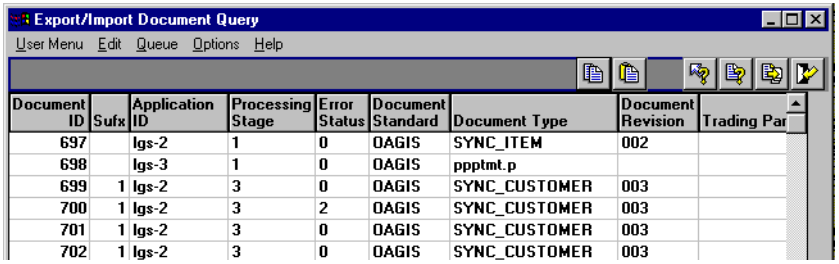


Fig. 12.2 Processing Stage and Error Status

In Export/Import Document Query, the processing stage for each document is noted only by code number:

Table 12.3
Processing Stage Codes

Code	Document Processing Stage	
	Export	**Import**
1	Published	Received
2	Mapped	Mapped
3	Sent	Processed
4	Acknowledged	Acknowledged

Error status codes are the same for import and export documents. The error status determines whether a document completed its last move between processing stages.

Table 12.4
Error Status Codes

Code	Error Status	Successful Move
0	Success	Yes
1	Warning	Yes
2	Error	No

Document Groups and Error Recovery

If a transmission fails while Q/LinQ is sending or receiving a document that is part of a series or batch group, resend that document and the remaining, unsent documents from the group. Do this before sending a new group or Q/LinQ might not be able to process the successfully transmitted part of the interrupted group. When the documents are resent, Q/LinQ appends the unsent portion of the group to the already received portion, restoring group integrity.

Exporting Data

This section discusses the export process flow, exporting sessions, export destinations, Send Export Documents (36.8.7) sessions, and how to use Send Export Documents.

Although there are version-specific differences in the export process flow (see Figure 12.3 and Figure 12.4), the general flow follows these points:

- The publishing API creates documents from MFG/PRO data.
- The mapping API maps the document data to the appropriate output format.
- The communication API exports the documents to the appropriate destination.

See Table 12.1, “Export Processing Stages,” on page 127.

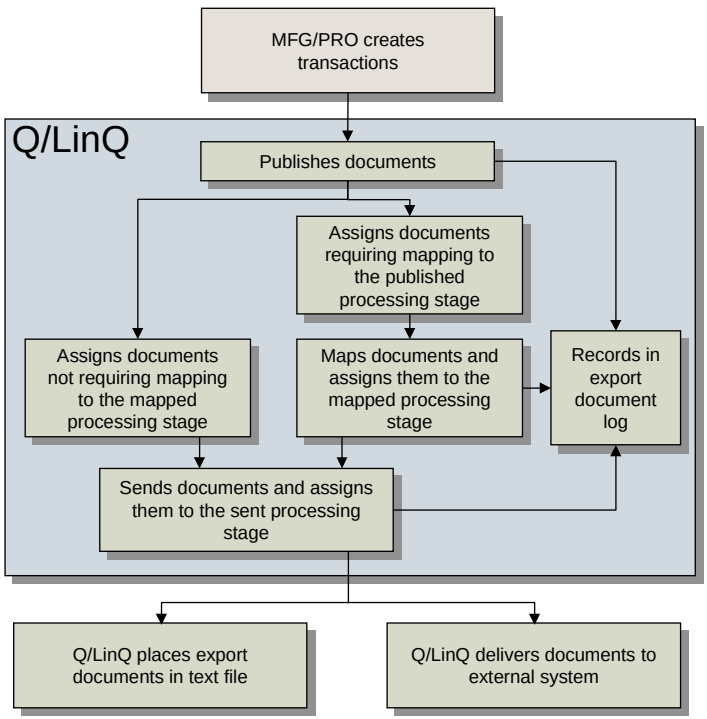
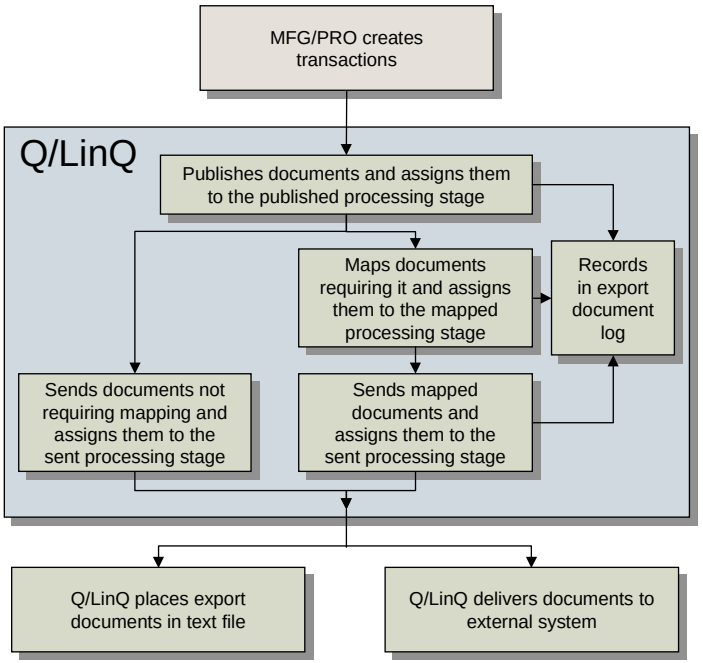


Fig. 12.3 Export Process Flow for 9.0

Fig. 12.4
Export Process Flow for MFG/PRO eB and eB2



Exporting to External Applications

The following are requirements for exporting documents to an external application:

See “Publishing API Reference” in *Technical Reference: Q/LinQ*.

- A user-written publishing API to assemble the MFG/PRO data into documents and publish the documents to Q/LinQ
- If necessary, a mapping API to map the data to the external application format
- An external application-specific messaging- or stream-communications API to export the data to the external application

See “Registering External Applications” on page 107.

- Registration in Register External Application (36.8.1.1; 36.8.1 in 9.0) to provide Q/LinQ with information about data mapping and communication to the external application

Forwarding to MFG/PRO

Q/LinQ can also send (forward) documents back into MFG/PRO. To forward documents back into MFG/PRO, enter the Q/LinQ system ID as the destination application ID in Send Export Documents (36.8.7). Define system ID in Q/LinQ Control (36.8.24).

The exported documents created using the forwarding feature operate like imported document when they are received from an external application, except that the sending application is the same MFG/PRO database that is importing the document.

Forwarding enables an MFG/PRO function to initiate another function outside its transaction scope. It is useful for initiating a transaction for a later Q/LinQ session to process—for example, asynchronous work flow within an MFG/PRO installation. Forwarding is similar to MFG/PRO batch processing, but is interactive rather than batch oriented.

Send Export Document Sessions

Send Export Documents (36.8.7) runs continuously throughout the MFG/PRO session if you enter:

- Any nonzero number for Pause Seconds Between Document Searches
- 0 (zero) for Disconnect Minutes After No Documents Remaining

Continuously searching for documents lets Q/LinQ map and send documents from MFG/PRO on an almost real-time background basis instead of through explicit batch runs.

Use the Interface Session Monitor (36.8.5) to pause, resume, or cancel the export session at any time. If the session is paused or canceled, a message displays in the Send Export Documents window. If a session is canceled (either through Interface Session Monitor, pressing End, or a system failure), you must manually delete the session record to release the external application so that other Q/LinQ sessions can connect to it. Delete session records using the delete option in Interface Session Monitor.

See “Editing Document Data” on page 147.

Q/LinQ supports multiple concurrent import and export sessions:

- One Q/LinQ export or import session can export to or import from one external application at a time.
- One Q/LinQ session can export to an external application while another Q/LinQ session simultaneously receives imports from the same application.
- Two concurrent Q/LinQ sessions cannot simultaneously send documents to the same external application. This limitation preserves the document sequence for processing by the external application.

You can start Send Export Documents as a batch process. Run the procedure by starting a batch Progress session. For example, use the `cron` command in UNIX. Use input redirected from an input file containing the UI-emulation data for the receiving application.

Note Sample UNIX batch scripts are provided with Q/LinQ in the directory: *`QlinqServerInstallDir`*`/samples/scripts`.

Using Send Export Documents

Send Export Documents (36.8.7) has two frames. In the first, specify processing parameters and the application; in the second, specify the documents to export.

The table and figures listed below describe the version-specific differences in assigning documents to export processing stages. Review these differences when making choices for the Action field.

- Table 12.1, “Export Processing Stages,” on page 127
- Figure 12.3, “Export Process Flow for 9.0,” on page 129
- Figure 12.4, “Export Process Flow for MFG/PRO eB and eB2,” on page 130

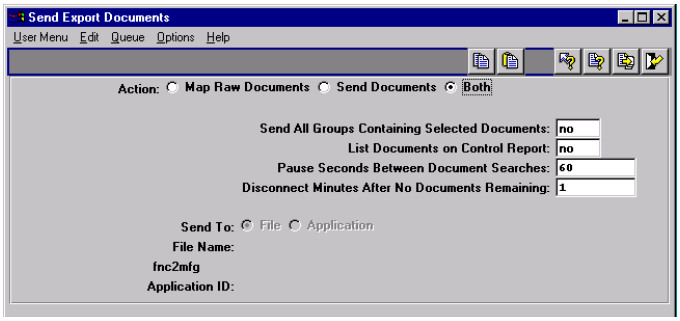


Fig. 12.5
Send Export Documents (36.8.7)

Action. Specify how documents should be processed.

- Select Map Raw Documents if the documents to export are not formatted for the external application.
- Select Send Documents if the documents are formatted for the external application. Documents must be in a format that the external application can read before Q/LinQ can send them.
- Select Both to combine mapping and sending in one step.

Send All Groups Containing Selected Documents. Enter Yes to process all documents in groups associated with the selected documents, even when these groups contain unselected documents. Q/LinQ expands your selection to include all the documents in the series or batch group. This is useful when you are mapping or sending a group of documents but do not remember all the document IDs in the group.

Enter No to skip the group if all its documents are not selected. After displaying a warning message, Q/LinQ does not process that group and moves on to the next group.

This field affects only documents sent in groups.

List Documents on Control Report. Enter Yes to include control information about each document on the control report. Enter No to have the report list only the number of document groups selected, processed, sent, containing errors, or skipped.

Control information includes:

- Document ID, group ID, application ID
- Published standard of the document that defines its structure and content after mapping, such as ANSI X12 or OAGIS

- Type of document, such as 850 for ANSI X12 or SYNC_SALESORDER for OAGIS BOD
- Trading partner ID
- Processing stage: published, mapped, sent, or acknowledged
- Error status: success, warning, failure

Select the printer or other output device to print the report in Output. Terminal and scrolling window are not valid options for output.

Pause Seconds Between Document Searches. Enter the time in seconds that Q/LinQ waits before looking in the export queue for the next document to export or process that meets the selection criteria.

Enter 0 to have Q/LinQ stop exporting after it has sent all pending documents and not wait for more published documents.

Enter any nonzero number to have Q/LinQ continuously monitor the export queue and send documents that meet the selection criteria. Q/LinQ stops when you end the session by pressing End.

Disconnect Minutes After No Documents Remaining. Enter the number of minutes Q/LinQ waits between sending a document and failing to find another document to send. If Q/LinQ fails to find another document that meets the selection criteria, it ends the procedure.

Enter 0 to have Q/LinQ repeatedly search for documents. Q/LinQ will not stop searching until you end the session by pressing End. Employ continuous searching to ensure processing of newly arrived data when Q/LinQ exports documents on a continuous basis. Exporting continuously is often done in a real-time, data-collection environment.

Send To. Select whether to send the documents to a text file or an external application for processing.

File Name. If you selected File in Send To, enter the name of the text file to save the documents in.

Application ID. If you selected Application in Send To, enter the identifier of the external application to receive the documents. This identifier is defined when the application is registered with Q/LinQ in Register External Application.

In the next frame, enter values for one or more parameters. Q/LinQ uses the parameters to select the documents to send.

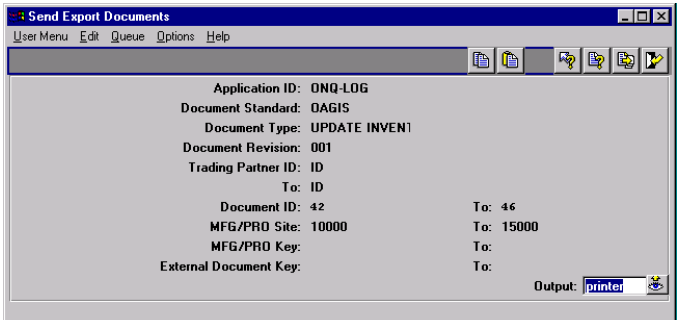


Fig. 12.6
Send Export Documents, Second Frame

Application ID. If you selected Application in Send To on the previous screen, this field displays the value you entered in Application ID. If you selected File, enter an application ID previously defined in Register External Application. Q/LinQ sends all the documents that are associated with this application and that meet the other selection criteria on this screen.

Document Standard. Enter the published standard of the document. Enter CIM if the documents are formatted for UI emulation for an MFG/PRO menu procedure or a custom Progress program. The standard describes the structure and content of the document. Examples of standards are EDIFACT, OAGIS, and ANSI X12.

Document Type. Enter the specific document type within the document standard; for example, the 850 transaction within the ANSI X12 standard or the OAGIS UPDATE_INVENTORY BOD.

Document Revision. Enter the revision number of the export document. For example, in the case of X12 EDI, this is the revision number of a specific X12 document definition. In the case of an OAGIS BOD, it is the three-digit number denoting the revision level of the BOD.

Trading Partner ID. Enter a range of trading partners to receive exported documents.

Depending on how trading partner is defined in Export Specification Maintenance, a trading partner is an actual business partner or a particular destination application that receives documents from middleware.

Document ID. Enter a range of unique document identifiers for Q/LinQ to export.

MFG/PRO Site. Optional. Enter a range of primary MFG/PRO host sites with which this MFG/PRO installation is associated. This field is used when Q/LinQ is deployed with multisite MFG/PRO implementations. The site code must be populated on documents that are published from MFG/PRO (exported) or from the external application (imported).

MFG/PRO Key. Enter a range of values that identify the documents to MFG/PRO users; for example, order-line number, item number, or customer number. For export documents, the API that publishes the documents from MFG/PRO to Q/LinQ supplies the MFG/PRO Key field value. For import documents, the API that publishes the documents from the external application to MFG/PRO supplies the value, which the API bases on an MFG/PRO value.

To use this selection criterion, the document field must have a value.

External Document Key. Enter a range of document identifiers assigned by the external application; for example, an external batch, transaction or document number. For export documents, the API that publishes the documents supplies the value, which the API bases on an external-system value. For import documents, the API that publishes the documents from the external application to MFG/PRO supplies the value.

To use this selection criterion, the document field must have a value.

Importing, Mapping, and Processing Data

This section covers import processing flow, using Receive Import Documents (36.8.9), and processing imported documents.

Although there are version-specific differences in the import process flow, see Figure 12.7 and Figure 12.8, the general flow follows these points:

See Table 12.2, “Import Processing Stages,” on page 127.

- The communication API imports documents from external applications into Q/LinQ.
- The mapping API maps the document data to MFG/PRO format.
- The processing API loads the data into MFG/PRO.

Although separate steps, both receiving and processing can run concurrently and continuously in separate sessions. This enables Q/LinQ to continuously receive documents and continuously process these newly received documents.

Since receiving usually occurs faster than processing, open several Q/LinQ processing sessions for each receiving session. Enter the same parameters for each processing session in order to process documents from a particular receiving session.

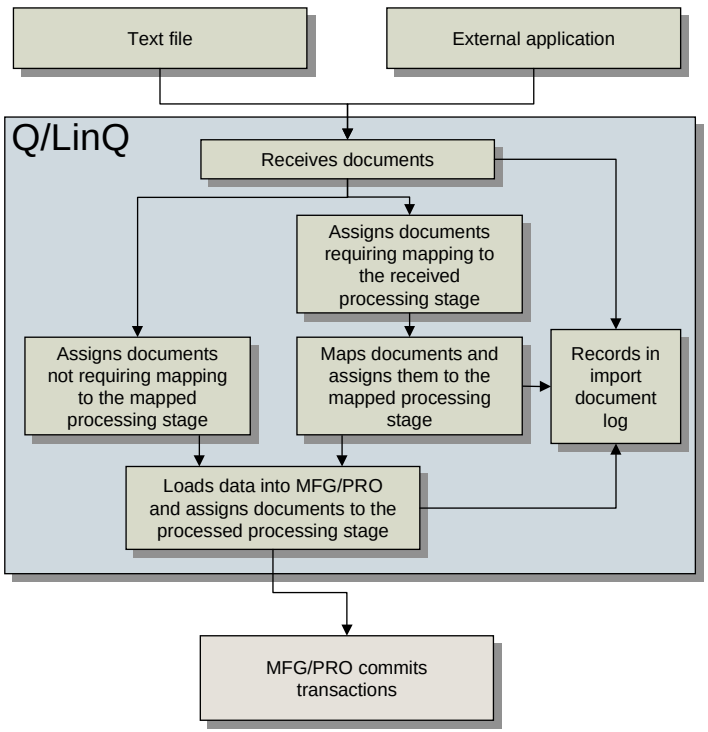
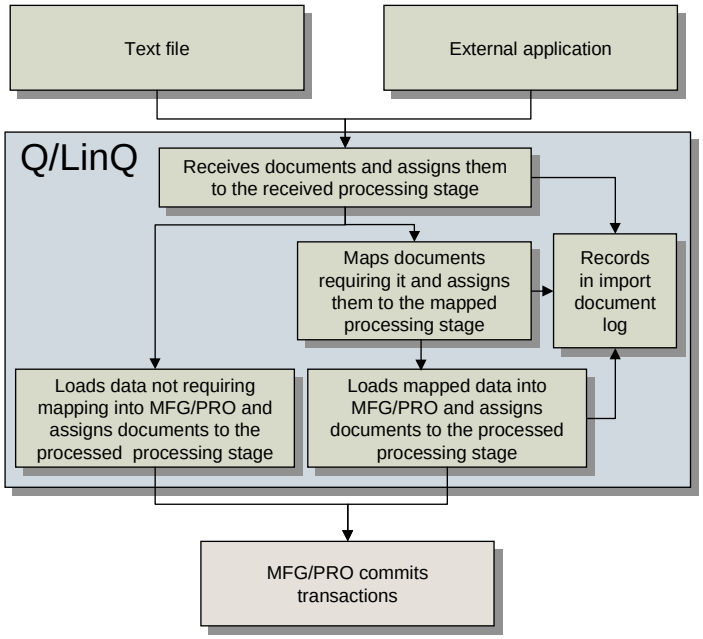


Fig. 12.7 Import Process Flow for 9.0

Fig. 12.8
Import Process Flow for MFG/PRO eB and eB2



Importing from External Applications

The following are requirements for importing documents from an external application:

See *Technical Reference: Q/LinQ*.

- An external application-specific adapter must exist to communicate with Q/LinQ through the communication API.
- The adapter must be registered in Register External Application (36.8.1.1; 36.8.1 in 9.0).
- The external application must be registered in Register External Application.

Q/LinQ can import data that is:

See Appendix B, “Text File Control Tags,” on page 167.

- An ASCII file including those formatted for UI emulation. A text file must use control tags to initiate and terminate the data for each transaction and provide control information for the Q/LinQ log records.
- The output of an application external to MFG/PRO that is written in Progress 4GL, or that can communicate with a C or Java API.

Receiving Import Documents

Import raw or mapped documents into Q/LinQ with Receive Import Documents (36.8.9).

Before you can import a document from an external application, it must be registered in Register External Application (36.8.1.1; 36.8.1 in 9.0). Registration provides Q/LinQ with information about communication with the external application and default values regarding the documents it generates that the external application might not provide.

If an Import Specification Maint (36.8.1.3; 36.8.3 in 9.0) record does not exist for a document, Q/LinQ uses the parameters defined in Register External Application.

- If the value for Document Standard for Import is OAGIS, Q/LinQ parses the BOD to get the document type and revision.
- If no standard is defined, Q/LinQ maps and processes the document according to the values in:
  - The Data Mapping Procedure field in the Default Data Mapping Parameters screen
  - The In MFG/PRO, In Registered Application, or Process Through User Interface fields in the Transaction Processing Programs frame of the Miscellaneous Defaults screen

Receive Import Documents can run continuously throughout the MFG/PRO session if you enter the following values:

- Any nonzero number for Polling Frequency If Session Paused
- 0 (zero) for Disconnect Minutes After No Documents Remaining

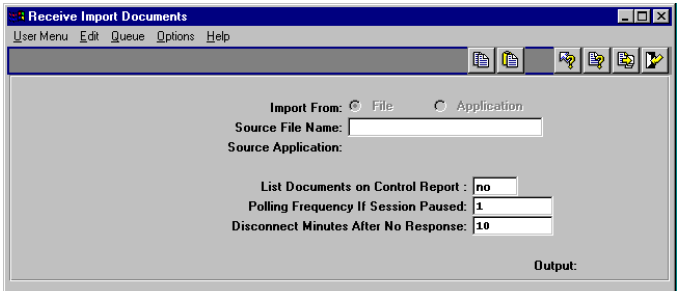
This lets Q/LinQ receive documents in the background almost in real time without using explicit batch runs. If Process Import Documents also continuously runs using the same Source Application field value, mapping and processing of imported documents occurs soon after Q/LinQ receives them.

You can pause, resume, or cancel the import session at any time using the Interface Session Monitor (36.8.5). If the session is paused or canceled, a message displays in Receive Import Documents.

You can start Receive Import Documents to receive from an application or file using a batch approach. Run the procedure by starting a batch Progress session. For example, use the `cron` command in UNIX.

Note Sample UNIX batch scripts are provided with Q/LinQ in the directory: *`QlinqServerInstallDir`*`/samples/scripts`.

Fig. 12.9
Receive Import Documents (36.8.9)



Import From. Select whether the documents to import are in a text file or the direct output of an application external to MFG/PRO. The application is the one generating the data or a message-oriented middleware application that communicates with the data-generating application.

If the documents are in a text file, Q/LinQ reads the data from the file.

If the documents are direct output of an application, they must go through an adapter. The adapter must be either another Progress procedure or a separate executable program running on UNIX or Windows. The Q/LinQ stream API includes procedure calls to Progress procedures. The Q/LinQ messaging API communicates with the adapter using TCP/IP sockets and C routines or Java classes.

Source File Name. Enter the path and file name of the file that contains the import documents. If you enter only a file name, Q/LinQ looks in the working directory.

Source Application. Enter the application whose processing output is the source of the import documents. The application must have been previously registered in Register External Application (36.8.1.1; 36.8.1 in 9.0).

If receiving from a file, Q/LinQ assumes that all the documents in the file came from this application, unless a document control tag overrides this value.

List Documents on Control Report. Enter Yes to include control information about each document on the control report. Enter No to have the report list only the number of document groups selected, processed, sent, containing errors, or skipped.

Control information includes:

- Document ID, group ID, application ID
- Published standard of the document that defines its structure and content after mapping, such as ANSI X12 or OAGIS
- Type of document, such as 850 for ANSI X12 or SYNC_SALESORDER for OAGIS BOD
- Trading partner ID
- Processing stage: published, mapped, sent, or acknowledged
- Error status: success, warning, failure

Select the printer or other output device to print the report in Output. Terminal and scrolling window are not valid options for output.

Polling Frequency If Session Paused. Enter the amount of time in seconds that Q/LinQ waits between polls of the external application API (specifically, its communication socket) when a Q/LinQ session is paused in Interface Session Monitor.

Q/LinQ stops polling when the value entered in Discontinue Minutes After No Response is exceeded.

Disconnect Minutes After No Response. Enter the maximum time in minutes for Q/LinQ to wait after receiving an import document before it ends the session. If you pause the session using Interface Session Monitor longer than the value entered in this field, the session automatically terminates.

Enter 0 (zero) to have Q/LinQ repeatedly poll for documents until you end the session by pressing End. Continuous import is often done in a message-oriented or real-time data-collection environment.

This field has no effect when Q/LinQ reads from a file.

Mapping and Processing Import Documents

After receiving import documents through Receive Import Documents (36.8.9), use Process Import Documents (36.8.10) to convert the document to the appropriate format. Convert the document by mapping it; then update the MFG/PRO database by processing it. You can map and process the document in two separate steps or together in one step.

Note Review the table and figures for more detailed information on import processing stages:

- Table 12.2, “Import Processing Stages,” on page 127
- Figure 12.7, “Import Process Flow for 9.0,” on page 137
- Figure 12.8, “Import Process Flow for MFG/PRO eB and eB2,” on page 138

The source application ID in each document can be different from the application the document was received from. Based on the source application ID, Q/LinQ processes the incoming data according to the application information in Register External Application (36.8.1.1; 36.8.1 in 9.0) or the document information in Import Specification Maint (36.8.1.3; 36.8.3 in 9.0). Q/LinQ can process an imported document in one of two ways:

Tip
UI-emulation mode resembles the CIM Interface method.

- Load the data using UI emulation.
- Invoke a gateway program that directly updates the MFG/PRO database.

Implement the UI-emulation file with Progress `input from` and `input through` constructs. The program logic and validations of the target MFG/PRO procedure (`@DESTPROC`) are used.

See Table B.2, “ADI Tags of the CIM Interface,” on page 173.

Q/LinQ can read and process unmodified ASCII data files formatted for the CIM Interface (36.15). See “Other MFG/PRO Documentation” on page 2 for references to CIM documentation.

Q/LinQ stores error or warning messages generated during this procedure in the import log. Use Export/Import Document Report (36.8.17) or Export/Import Document Query (36.8.16) to view the log.

Use Interface Session Monitor (36.8.5) to pause, cancel, or resume the processing or mapping. When a session is paused or canceled, Interface Session Monitor displays the statistics of the processing or mapping session; for example, the document ID of the last document processed.

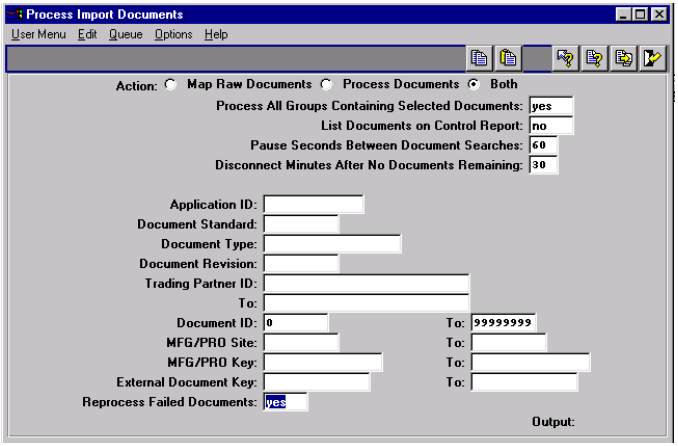


Fig. 12.10 Process Import Documents (36.8.10)

Process Import Documents runs continuously throughout the MFG/PRO session if you enter:

- Any nonzero number for Pause Seconds Between Document Searches
- 0 (zero) for Disconnect Minutes After No Documents Remaining

This lets Q/LinQ map or process documents on an almost real-time background basis instead of through explicit batch runs. If Receive Import Documents also runs continuously using the same Source Application field value, mapping and processing of imported documents occurs soon after Q/LinQ receives them.

To increase the number of documents that can be processed, open this procedure concurrently with identical or overlapping selection criteria in multiple MFG/PRO log-in sessions. Each procedure takes the next unprocessed document from the Q/LinQ import queue using first-in-first-out order, as it completes a document. For a batch or series group, one Q/LinQ session locks all the group documents and processes them in sequence.

If you set Reprocess Failed Documents to No, Q/LinQ only attempts to process documents within the other selection criteria that have not

previously failed. This offers you more flexibility in fixing problems and can save processing time.

You can start Process Import Documents in a batch Progress session. For example, use the `cron` command in UNIX.

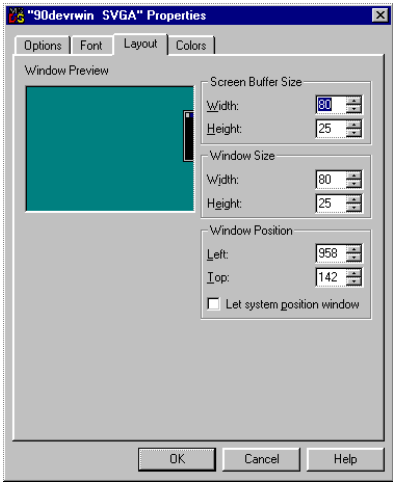
Note Sample UNIX batch scripts are provided with Q/LinQ in the directory: *`QlinqServerInstallDir`*`/samples/scripts`.

Repositioning the Command Window

When running Process Import Documents interactively in the Windows interface, the system displays a DOS command window. By default, this window displays in the center of the screen and may overlap the program window. To have the command window display in a less obtrusive location, do the following:

1. Right-click the command window title bar and select Properties.
2. In the Properties window, click the Layout tab.
3. Clear the Let system position window option.
4. Using the Window Preview display as a guide, enter new values in the Left and Top fields to reposition the window in a more appropriate location. See Figure 12.11 for an example.

Fig. 12.11
Properties Window



5 Click OK to apply the new settings.

6 When the Apply Properties To Shortcut Window displays, select Modify shortcut which started this window. This ensures that the command window displays in the specified location each time you run Process Import Documents.

Monitoring Q/LinQ Sessions

Use Interface Session Monitor (36.8.5) to pause, resume, or cancel a Q/LinQ session or to view its status.

Q/LinQ assigns a unique session ID to every session and associates the documents in the session with that ID. The session ID functions as a lock that prevents documents from simultaneous processing by two concurrent sessions.

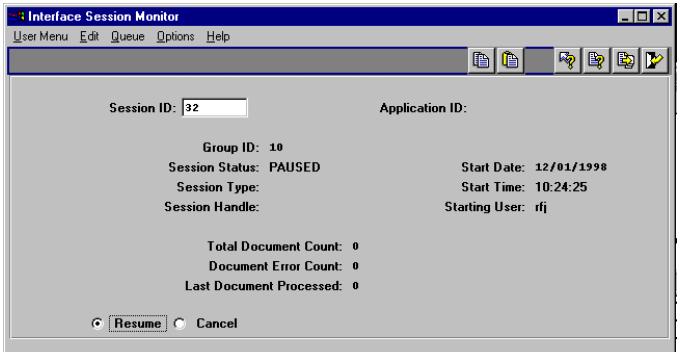


Fig. 12.12 Interface Session Monitor (36.8.5)

Enter a Session ID and press Go. The monitor displays the number of documents sent or received, the number of errors encountered, and the document ID of the last document sent, received, or processed.

You must manually delete records for canceled sessions. This releases the external application so that other Q/LinQ sessions can connect to it. After canceling a session, enter its Session ID and press Go. Delete the session records by selecting Delete Records in the next displayed screen.

Sessions can be canceled in more than one way:

- Using the Interface Session Monitor
- By pressing End or Exit from MFG/PRO

- A system failure

See “Starting and Restarting Q/LinQ” on page 118.

Note If a system failure canceled the session, stop and restart the MOM adapters before restarting Q/LinQ.

Editing Documents, Data, Control Tags

Editing CIM Documents

Use Debug CIM Document (36.8.11) to check for errors in a CIM-formatted import document. The procedure runs the data from one CIM document through the MFG/PRO program specified in the @DESTPROC value (see Table B.1 on page 169), displaying it on the screen.

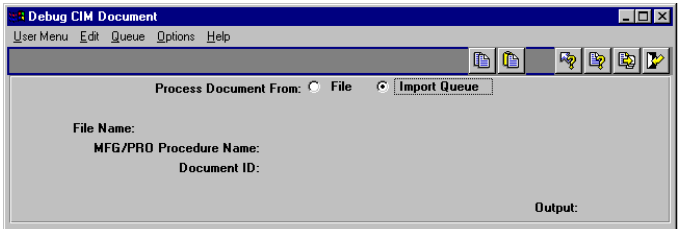
If there are no errors, you can commit the transaction; this is equivalent to processing the document through Process Import Documents (36.8.10). If there are errors, open Dump Export/Import Doc for Edit (36.8.13) to correct them, then reopen Debug CIM Document.

This program captures every MFG/PRO screen display that would have resulted had the data been entered manually. Because MFG/PRO displays the same screen data many times, generally once each time Go is pressed, the output of this program contains redundant screen-display data. Review all the output to determine which screens Q/LinQ entered the CIM data into, as well as any error or warning messages.

See “Transaction Document Groups” on page 84.

Since a document in a series or batch group must be processed in the appropriate sequence, you cannot commit the transaction after successfully processing it in Debug CIM Document. Use Process Import Documents to process all grouped documents.

Fig. 12.13 Debug CIM Documents (36.8.11)



Debug documents formatted as CIM Interface transactions either before or after importing them with Receive Import Documents (36.8.9).

To debug a CIM document after importing it, enter the document ID.

To debug a CIM document before importing it, enter both the file name and the MFG/PRO procedure that is to process it. The document must be in a text file and have no CIM Q/LinQ control tags, such as @@GROUP, @@DOC, or @@BATCHLOAD. Debug CIM Document processes only one document at a time. When reading from a multi-file document, use Enter to scroll to the next document and process each in sequence.

Note The meanings of group and group ID in the CIM Interface are not equivalent to those of group and group ID as they are used in this book.

Debug CIM Document cannot debug ADI (application data interface) formatted documents from MFG/PRO Object-Based Component Model (OBCM) programs.

Note OBCM programs no longer exist in MFG/PRO eB2.

Editing Document Data

If an import or export document contains errors that prevent MFG/PRO or an external application from processing it or MFG/PRO from sending it, you can edit the document data rather than rerunning the transaction that produced the document. Dump Export/Import Doc for Edit (36.8.13) dumps the raw—not mapped—data of *one* document to an ASCII file. Edit using any text editor.

If the error is in an import document, consider making the correction at the source—the sending application—and resending it to Q/LinQ.

Warning Be careful when using this program; you can easily violate data security or integrity.

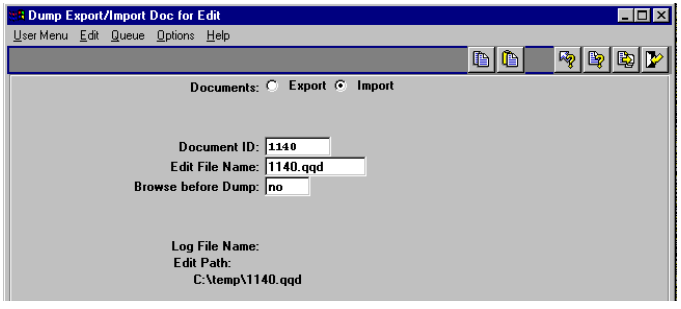


Fig. 12.14
Dump Export/ Import Doc for Edit (36.8.13)

See “Editing Document Control Tags” on page 148.

Note Use Document Control Tag Maint (36.8.15) to edit control tag values.

Enter the document ID for an import or export document and a file name for it. If you do not know the document ID, enter Yes in Browse before Dump, and a browse of document IDs and their associated raw data displays. Select a document and press Go. The Log Header frame displays.

Q/LinQ dumps the document to a text file in the path you enter. You can edit the text file using any text editor.

When you have finished correcting the document, reload it using Reload Edited Export/Import Doc (36.8.14). During the reload, the program deletes any existing corresponding raw or mapped documents. However, this program lets you save the original document data as well as the corrected data in a separate log file. You can also delete the text file used to edit the document.

After the document is reloaded, Q/LinQ clears the document error status and processing-stage fields in the document log. You can retry mapping and exporting the document using Send Export Documents (36.8.7) or reprocessing the import document with Process Import Documents (36.8.10).

Use this program to correct document contents for one document at a time. To dump a document to a file for export, use Send Export Documents. To copy a document, edit it, and load it as a new document, use Dump Export/Import Docs to File (36.8.18).

Q/LinQ stores any errors or other messages generated during processing in the import or export document log. Use Export/Import Document Report (36.8.17) to view the log.

Editing Document Control Tags

When importing documents in files, the documents must include the tags in Table B.1 on page 169. These control tags identify the document to Q/LinQ.

Use Document Control Tag Maint (36.8.15) when a document requires alteration of its control-tag values; for example, if a document returns mapping errors due to an incorrect document type value.

If many documents have the error, consider fixing the control-tag error at the origin, where the documents were originally built or mapped, rather than altering them one at a time.

Before you import or export the corrected documents, delete the incorrect versions.

Warning Correcting the documents at the source is the recommended method. Use this program carefully.

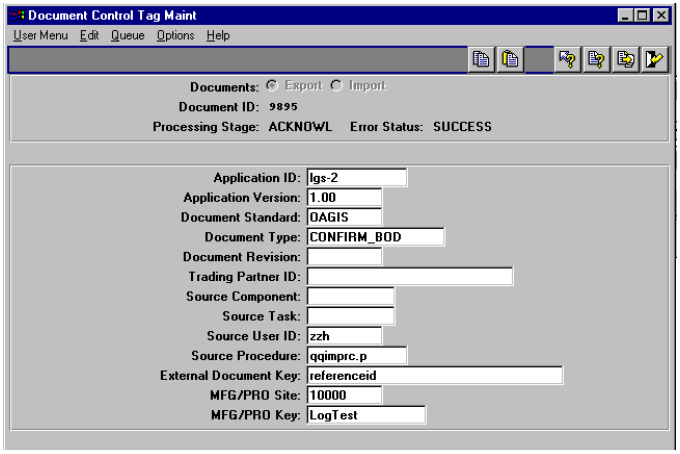


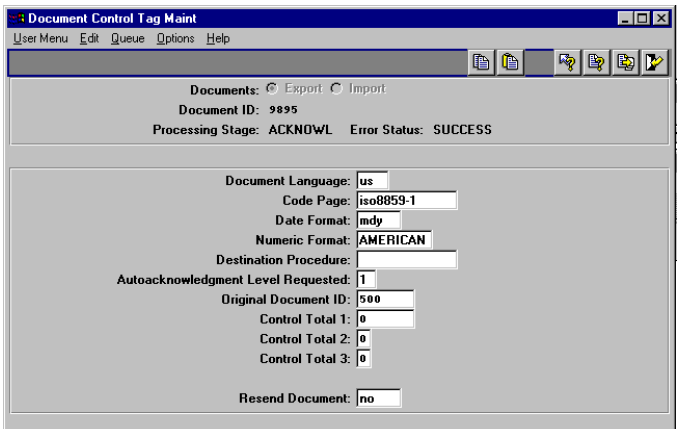
Fig. 12.15
Document Control Tag Maint (36.8.15)

Select whether the document to fix is for export or has been imported. Enter the document ID. If the document has a suffix because Q/LinQ mapped it to multiple documents, enter it in the unlabeled field to the right of the Document ID field. This field is only displayed when required.

See “One-to-One Mapping” on page 98.

Q/LinQ displays the processing stage and error status of the document you selected. Press Go to move to the next frame, which displays the current values for you to edit. Press Go to display more values in another frame. If you need to resend an export document, enter Yes in Resend Document.

Fig. 12.16
Document Control Tag Maint, Second Frame



Resend Document. Enter Yes to enable export of this document again. Use Send Export Document to export it and any associated documents. After Q/LinQ resends the document, it resets this value to No.

Note If the document has been deleted from Q/LinQ, you cannot export it again. You must create another document by rerunning the original transaction that supplied the document data.

Dumping Data to a File

Use Dump Export/Import Docs to File (36.8.18) to transfer a range of documents to a text file. This is most useful for copying existing documents, altering them, then loading the new documents into Q/LinQ. Load the new documents using Receive Import Documents (36.8.9).

Warning Do not use this procedure to dump documents for transport to external applications. Since it is not a Q/LinQ export process, Q/LinQ cannot track the documents and does not record the dumped documents as sent in the log.

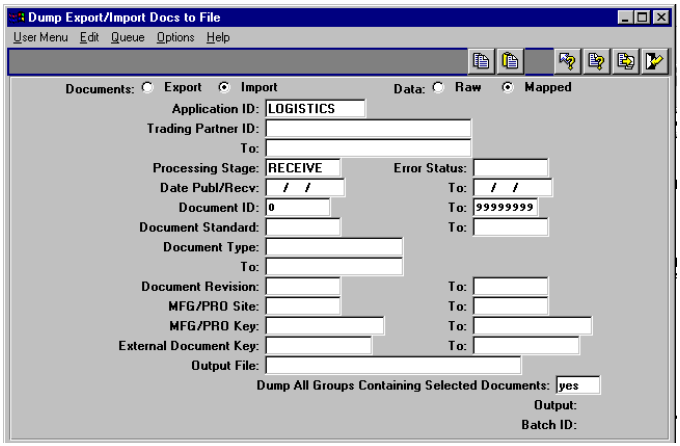


Fig. 12.17
Dump Export/ Import Docs to File (36.8.18)

To export documents from MFG/PRO to other applications, use Send Export Documents (36.8.7). To dump documents to archives, use Export/ Import Doc Delete/Archive (36.8.23).

Q/LinQ stores the original data of import and export documents in a file. You can dump the raw document data or the mapped data. The output of this procedure is a text file of a set of data lines wrapped in header lines. Identifier values in the header describe key document attributes, such as the document type and the source or destination application. The header also marks the beginning of groups and defines their type: series, batch, or individual.

Note Q/LinQ dumps raw data if mapped data is requested but not present.

Generating Reports

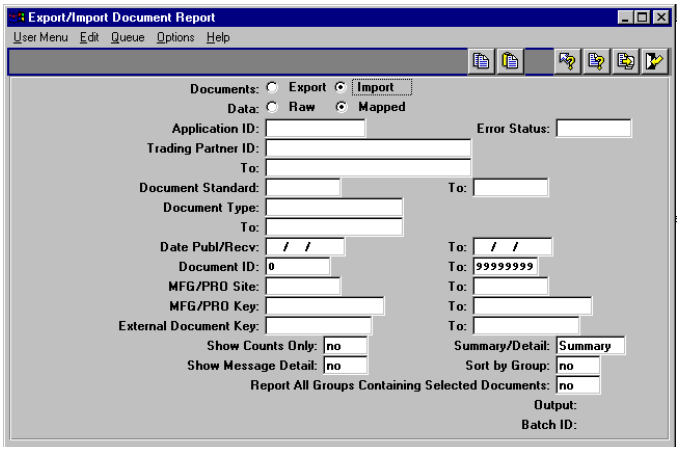
Export/Import Document Report (36.8.17) creates various summary or detail reports. Use the majority of the fields to specify which documents to include in the report:

- Imported or exported documents
- Raw data
- Mapped data
- Documents imported from or exported to a particular external application

- Documents having a particular error status
- Documents received or exported within a certain date range
- Documents whose document ID values fall within a range

Note In MFG/PRO 9.0, Q/LinQ reports raw data when mapped data does not exist for the specified documents.

Fig. 12.18
Export/Import Document Report (36.8.17)



Use the remaining fields to specify which information to report:

Show Counts Only. Enter Yes to report only total counts of documents successfully and unsuccessfully processed.

Show Message Detail. Enter Yes to include return codes, messages, and severity levels in the report.

Summary/Detail. Enter Detail to print document log header details, the actual import or export data, and messages and their severity level.

Default is Summary, which prints only the document log header details; for example, document, application, and trading partner IDs; document standard, type, and revision; processing stage; and error status.

Sort by Group. Enter Yes to have the documents sorted by group in the report. There is no effect if the documents are not part of a group.

Instead of labels, the report shows numeric values for some characteristics. Table 12.5 lists the characteristics and their numeric values.

Table 12.5 Report Labels and Numeric Values

Characteristic	Label	Numeric Value
Processing stage	RECEIVED	1
	PUBLISHED	1
	MAPPED	2
	PROCESSED	3
	SENT	3
	ACKNOWLEDGED	4
Error status	INFO	0
	WARNING	1
	ERROR	2
Acknowledgment level	NONE	0
	ERROR-ONLY	1
	ALL	2

Deleting or Archiving Documents

Q/LinQ provides a procedure for managing disk space by deleting or archiving documents and their logs, which contain header information. If your company exchanges much data using Q/LinQ, you should probably run Export/Import Doc Delete/Archive (36.8.23) daily. There is no reason to save copies of document data and messages created by Q/LinQ after the data has been processed or messages sent.

Important The number of documents in the Q/LinQ log tables can grow very quickly in high-volume environments. Also, processing database delete transactions can be time consuming. Therefore, frequently delete or archive documents.

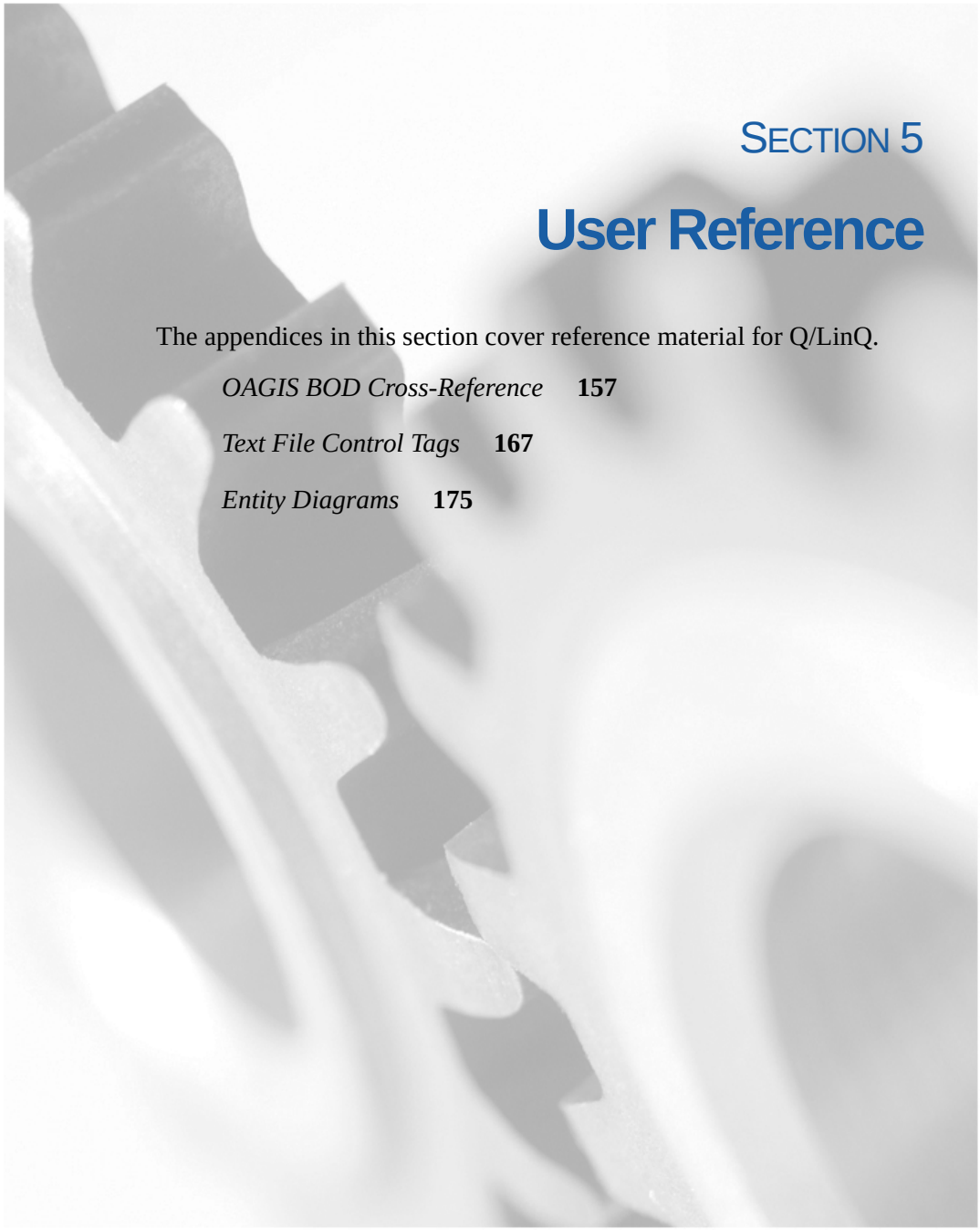
This procedure deletes the documents you select from Q/LinQ tables. If you choose to archive the documents, they are archived and deleted. It has no effect on the processed transactions in MFG/PRO or an external application that used data from the documents.

You can start Export/Import Doc Delete/Archive using a batch approach. Run the procedure by starting a batch Progress session. For example, use the `cron` command in UNIX.

Note Sample UNIX batch scripts are provided with Q/LinQ in the directory: *`QlinqServerInstallDir`*`/samples/scripts`.

Archive or delete the following:

- Raw or mapped import documents
- Export documents that have been published, mapped, sent, or acknowledged
- Documents imported from or exported to the external application entered in Application ID
- Documents whose structure and content follow the standard entered in Document Standard, such as EDIFACT, CIM, or OAGIS, and a type within that standard
- Documents to or from a particular trading partner
- Documents having a particular error status
- Documents received or exported within a certain date range
- Documents whose document ID values fall within a range



SECTION 5

User Reference

The appendices in this section cover reference material for Q/LinQ.

OAGIS BOD Cross-Reference **157**

Text File Control Tags **167**

Entity Diagrams **175**

Appendix A

OAGIS BOD Cross-Reference

This chapter provides detailed reference information on the format of the OAGIS business object documents (BODs) used by Q/LinQ for autoacknowledgments.

Overview **158**

Conventions **158**

BOD Stylesheet Specifications **159**

Control Area Fields Cross-Reference Tables **159**

Q/LinQ CONFIRM_BOD BODs **162**

CONFIRM_BOD Example **165**

Overview

This chapter cross-references the OAGIS BODs used by Q/LinQ to send and receive autoacknowledgments. They are in Table A.4 on page 162 and Table A.5 on page 163. These BODs support XML (extensible markup language), the meta-data standard supported by the World Wide Web Consortium.

Conventions

The CNTROLAREA section of BODs is not shown in the BOD tables; it is included in separate tables; see Table A.2 on page 159 and Table A.3 on page 160.

Q/LinQ can derive all control area fields from OAGIS BODs. Only the @@DOC, @DATA, and @@DOCEND Q/LinQ control tags are required for OAGIS BODs. In the tables, an at sign (@) prefaces Q/LinQ document control tags in the MFG/PRO field name column. For more information about Q/LinQ control tags, see Appendix B, “Text File Control Tags,” on page 167.

Bracketed numbers in MFG/PRO field names indicate the field is indexed. For example, the MFG/PRO field so_slspsn, which maps to the SALESPERSN field in the SYNC_SALESORDER BOD, is listed as so_slspsn[1], so_slspsn[2], so_slspsn[3], so_slspsn[4], indicating it is indexed up to four times.

Table A.1 lists XSLT stylesheets used by CONFIRM_BOD.

BOD Stylesheet Specifications

Table A.1 CONFIRM_BOD BOD Stylesheets

BOD	Rev	Functions	Import		Export	
			Table Location	Map Spec Files	Table Location	Map Spec Files
CONFIRM_BOD	002	• Imports processing acknowledgment • Exports processing acknowledgment	Table A.4 on page 162	`coBo2i.xsl`	Table A.5 on page 163	`coBo2e.xls`

Control Area Fields Cross-Reference Tables

Table A.2 Import BODs CNTROLAREA

BOD Field Name	Required for MFG/PRO?	MFG/PRO Field Name	MFG/PRO Field Label or Description	Comments
CNTROLAREA				
	See comment	@DOCSTD	Document standard	If the Q/LinQ registry entry specifies OAGIS as the application's standard, this tag is not needed. Otherwise, it is required.
VERB		@DOCTYP	Document type	
NOUN		@DOCTYP	Document type	
REVISION		@DOCREV	Document revision	
LOGICALID	Optional	@SYSID	Sending application ID	Not implemented
COMPONENT	Optional	@SRCCOMP	Source component	The component generating the document
TASK	Optional	@SRCTASK	Source task	
REFERENCEID	Optional	@EXTDOCKEY	External document key	Used for acknowledgments
CONFIRMATION	Optional	@ACKLVLREQD	Acknowledgment level	Default is no acknowledgment.
LANGUAGE	Optional	@DOCLNG	Document language	
CODEPAGE	Optional	@CODEPG	Code page	Default is the MFG/PRO code page default.
AUTHID	Optional	@SRCUSERID	Source user ID	

Table A.2 Import BODs CNTROLAREA

BOD Field Name	Required for MFG/PRO?	MFG/PRO Field Name	MFG/PRO Field Label or Description	Comments
DATETIME(CREATION)				Creation time/date information
YEAR MONTH DAY	Optional	@DATECREATE		
HOUR MINUTE SECOND	Optional	@TIMECREATE		
TIMEZONE	Optional	@TIMEZONE		qqc_timezone

Table A.3 Export BODs CNTROLAREA

MFG/PRO Field Name	MFG/PRO Field Label or Description	BOD Field Name	Comments
CNTROLAREA			
@DOCID	Document ID	—	elg_doc_id
@APPID	Application ID	REFERENCEID	Source application
@SYSID	System ID	LOGICALID	qqc_sys_id
@DOCSTD	Document standard	—	OAGIS
@DOCTYP	Document type	[For example, CONFIRM_ISSUE]	Verb_Noun
@DOCREV	Document revision	REVISION	The three-digit revision level; for example, 002
@ACKLVLREQD	Acknowledgment level	CONFIRMATION	elg_ack_lvl_reqd
@DOCLNG	Document language	LANGUAGE	elg_doc_lng
@CODEPG	CodePage	CODEPAGE	elg_code_pg
@SRCCOMP	Source component	COMPONENT	elg_src_comp

Table A.3 Export BODs CNTROLAREA

MFG/PRO Field Name	MFG/PRO Field Label or Description	BOD Field Name	Comments
@SRCTASK	Source task	TASK	elg_src_task
@SRCUSERID	Source user ID	AUTHID	elg_userid
DATETIME(CREATION)			Creation time/date information
@DATEPUBL		YEAR MONTH DAY	Derived from elg_date_publ
@TIMEPUBL		HOUR MINUTE SECOND	Derived from elg_time_publ
@TIMEZONE		TIMEZONE	qqc_timezone

Q/LinQ CONFIRM_BOD BODs

Table A.4 CONFIRM_BOD Import (Page 1 of 2)

BOD Field Name	Required for MFG/PRO?	MFG/PRO Field Name	MFG/PRO Field Label or Description	Comments
CONFIRM Data Type				
	See comment	elg_doc_std (@DOCSTD)	Document standard	If the application is registered with Q/LinQ and specifies OAGIS as the standard, this tag is not needed.
VERB	Required	elg_doc_typ (@DOCTYP)	Document type	CONFIRM. First portion of doc type, before underscore.
NOUN	Required	elg_doc_typ (@DOCTYP)	Document type	BOD. Second portion of doc type, after underscore.
REVISION	Required	elg_doc_rev (@DOCREV)	Document revision	002
LOGICALID		qqc_sys_id	Sending application ID	
COMPONENT	Optional	elg_src_comp (@SRCCOMP)	Source component	
TASK	Optional	elg_src_task (@SRCTASK)	Source task	
REFERENCEID	Required	elg_doc_id	Q/LinQ document identifier	Q/LinQ-assigned document identifier for the export document being acknowledged.
CONFIRMATION	Optional	elg_acklvlreqd (@ACKLVLREQD)	Acknowledgment level	Default = no acknowledgment
LANGUAGE	Optional	elg_doc_lng (@DOCLNG)	Document language	
CODEPAGE	Optional	elg_code_pg (@CODEPG)	Code page	Default = MFG/PRO code page default
AUTHID	Optional	elg_src_userid (@SRCUSERID)	Source user ID	Source user ID of the document being acknowledged.

Table A.4 CONFIRM_BOD Import (Page 2 of 2)

BOD Field Name	Required for MFG/PRO?	MFG/PRO Field Name	MFG/PRO Field Label or Description	Comments
DATETIME(CREATION)	Optional	elg_date_publ (@DATECREATE) elg_date_publ (@TIMECREATE) qqc_timezone (@TIMEZONE)		Creation time/date information.
STATUSLVL		elg_ack_stat		Indicates whether the export document was processed at its destination with errors, warnings, or neither.
CONFIRMMSG Data Type				
DESCRIPTN		emg_return_txt		Description of the error or warning condition. The description is limited to 75 characters; additional characters are truncated.
REASONCODE		emg_return_code		Return code identifying the error or warning condition.

The MFG/PRO field names in Table A.5 are from the original imported document that is being acknowledged.

Table A.5 CONFIRM_BOD Export (Page 1 of 2)

MFG/PRO Field Name	MFG/PRO Field Label or Description	BOD Field Name	Comments
CONFIRM Data Type			
ilg_app_id	Sending application ID	LOGICALID	Source application ID of the document being acknowledged.
ilg_src_comp	Source component	COMPONENT	Source component of the document being acknowledged.
ilg_src_task	Source task	TASK	Source task of the document being acknowledged.
ilg_ext_doc_key	External document key	REFERENCEID	Document key assigned by the external application to the document being acknowledged.

Table A.5 CONFIRM_BOD Export (Page 2 of 2)

MFG/PRO Field Name	MFG/PRO Field Label or Description	BOD Field Name	Comments
ilg_ack_lvl_reqd	Acknowledgment level	COMFIRMATION	Acknowledgment required indicator from the document being acknowledged.
ilg_doc_lng	Document language	LANGUAGE	Document language of the document being acknowledged.
ilg_code_pg	Code page	CODEPAGE	Codepage of the document being acknowledged.
ilg_userid	Source user ID	AUTHID	Source user ID of the document being acknowledged.
ilg_date_create, ilg_time_create, ilg_timezone	Publish time/date information	DATETIME (CREATION)	Creation data-time of the document being acknowledged.
ilg_err_stat	Error status	STATUSLVL	Error status of the document being acknowledged.
ilg_mfgpro_key	Document ID	ORIGREF	MFG/PRO identifier of the document being acknowledged.
CONFIRMMSG Data Type			
img_return_txt		DESCRIPTN	Description of the error/warning condition.
img_return_code		REASONCODE	Return code identifying the error/warning condition.
ilg_doc_sufx		USERAREA QAD.CONFIRM_BOD _002.CONFIRMMSG. USERAREA QAD.DOCSUFX	Sequentially assigned document suffix identifying the imported document within a multi-part BOD, that is, a BOD with multiple occurrences of DATAAREA, where the error/warning condition was detected.

CONFIRM_BOD Example

```
<?xml version="1.0" standalone = "no"?>
<!DOCTYPE CONFIRM_BOD_002 SYSTEM "002_confirm_bod_002c.dtd">
<CONFIRM_BOD_002>
    <CNTROLAREA>
        <BSR>
        <VERB>CONFIRM</VERB>
        <NOUN>BOD</NOUN>
        <REVISION>002</REVISION>
        </BSR>
        <SENDER>
        <LOGICALID>OnQ</LOGICALID>
        <COMPONENT>BUSNENTITY</COMPONENT>
        <TASK>MAINTENANC</TASK>
        <REFERENCEID>0001</REFERENCEID>
        <CONFIRMATION>0</CONFIRMATION>
        <LANGUAGE>us</LANGUAGE>
        <CODEPAGE>64</CODEPAGE>
        <AUTHID>QAD</AUTHID>
        </SENDER>
        <DATETIME qualifier = "CREATION">
        <YEAR>1998</YEAR>
        <MONTH>11</MONTH>
        <DAY>20</DAY>
        <HOUR>00</HOUR>
        <MINUTE>00</MINUTE>
        <SECOND>00</SECOND>
        <SUBSECOND>0000</SUBSECOND>
        <TIMEZONE>-0600</TIMEZONE>
        </DATETIME>
    </CNTROLAREA>
<DATAAREA>
    <CONFIRM_BOD>
        <CONFIRM>
            <CNTROLAREA>
            <BSR>
            <VERB>CONFIRM</VERB>
            <NOUN>BOD</NOUN>
            <REVISION>002</REVISION>
            </BSR>
            <SENDER>
            <LOGICALID>OnQ</LOGICALID>
            <COMPONENT>BUSNENTITY</COMPONENT>
            <TASK>MAINTENANC</TASK>
            <REFERENCEID>0002</REFERENCEID>
            <CONFIRMATION>0</CONFIRMATION>
            <LANGUAGE>us</LANGUAGE>
            <CODEPAGE>64</CODEPAGE>
            <AUTHID>QAD</AUTHID>
            </SENDER>
            <DATETIME qualifier = "CREATION">
            <YEAR>1998</YEAR>
            <MONTH>11</MONTH>
            <DAY>20</DAY>
            <HOUR>00</HOUR>
            <MINUTE>00</MINUTE>
```

```
            <SECOND>00</SECOND>
            <SUBSECOND>0000</SUBSECOND>
            <TIMEZONE>-0600</TIMEZONE>
            </DATETIME>
            </CNTROLAREA>
            <STATUSLVL></STATUSLVL>
            <DESCRIPTN>CONFIRM BOD IMPORT</DESCRIPTN>
            <ORIGREF>RCPT#123</ORIGREF>
            <USERAREA>
            <QAD.CONFIRM_BOD_002.CONFIRM.USERAREA>
            <QAD.USER1>10000</QAD.USER1>
            </QAD.CONFIRM_BOD_002.CONFIRM.USERAREA>
            </USERAREA>
        </CONFIRM>
        <CONFIRMMSG>
            <DESCRIPTN>SYSTEM X SCREEN</DESCRIPTN>
            <REASONCODE>0001</REASONCODE>
        </CONFIRMMSG>
    </CONFIRM_BOD>
</DATAAREA>
</CONFIRM_BOD_002>
```

Appendix B

Text File Control Tags

Introduction **168**

Q/LinQ Tags **168**

ADI Tags **173**

Application Control Tags **173**

Introduction

Control tags provide Q/LinQ with information about text files that is similar or identical to the information that Register External Application (36.8.1.1; 36.8.1 in 9.0) provides regarding external applications.

The file can be any type of text file, including an OAGIS BOD or a CIM Interface-formatted transaction.

This appendix contains tables listing the control tags:

- Tags for imported text files:
  - Table B.1 lists the Q/LinQ tags for a text file.
  - Table B.2 lists the ADI tags of the CIM Interface, which Q/LinQ can read and use instead of the Q/LinQ tags.
- Tags for direct data exchange; that is, data imported from an application rather than a text file. Table B.3 lists the Q/LinQ control tags available to use when importing directly from an application process.

Q/LinQ Tags

Construct the files using the tags in the order presented in Table B.1. The tags initiate and terminate the data for each transaction and provide control information for Q/LinQ log records.

The `@@DOC` tag marks the beginning of one document (or transaction). If the document is one of a series or batch group, the `@@GROUP` tag marks the beginning of the group.

Table B.1
Text File Control-Tag Format

Tag	Purpose	Req
@@GROUP <BATCH or SERIES>	Indicates the document sequence dependency: how it should be grouped when processed with others created before or after it. • Series groups are processed in the same order they are received. If one document in the group has an error, the documents processed before it are committed, but the document with the error and the documents following it are not processed. • Batch groups are also processed in the same order received, but if one has an error, no document in the batch is committed. Documents already processed are rolled back. • Leave blank if document is independent from any other and relative processing sequence is not significant.	N
@@DOC	Required. Signals Q/LinQ that this is the beginning of the control data. This line has no control-tag value.	Y
@APPID <value>	The source or destination application.	N
@APPVER <value>	The version of the source or destination application that the document came from or goes to.	N
@DOCSTD <value>	Required for documents that have Document Standard values in Import Specification Maint. The published standard of the document. The standard describes the structure and content of the document. Examples of standards are EDIFACT, OAGIS, and ANSI X12. This field is optional if the document is the output of custom legacy applications and does not follow a published, revision-controlled standard such as OAGIS.	Y
@DOCTYPE <value>	Required for documents that have Document Type values in Import Specification Maint. The specific document type within the document standard after Q/LinQ maps it. Examples include the 850 transaction within the ANSI X12 standard or the OAGIS verb and noun of the BOD, separated by an underscore; for example, UPDATE_INVENTORY.	Y
@DOCREV <value>	Required for documents that have Document Revision values in Import Specification Maint. Revision number of the exported document after mapping by Q/LinQ.	Y

Table B.1 — *Text File Control-Tag Format* — (Page 1 of 4)

Tag	Purpose	Req
@TRADPTRID <value>	Required for documents that use this field in Import Specification Maint. The trading partner that sent the document or receives it.	Y
@DESTPROC <value>	The destination procedure. If it is an import document, the MFG/PRO procedure that processes it. It is a Progress 4GL procedure unless the import document is a CIM document. Then, the procedure is the MFG/PRO online application program whose input data stream Q/LinQ emulates. If it is an export document, this is the procedure at the destination that processes the document.	N
@SRCCOMP <value>	The name of the registered application component or module that generates documents for Q/LinQ. This is usually a major business function or module, such as AR or Receiving. Use this field for reference when storing this kind of historical information is important. In particular, use it with OAGIS BODs where it maps to COMPONENT in the SENDER segment of the CNTROLAREA.	N
@SRCTASK <value>	The task or activity in the external application that generates documents for Q/LinQ. For example, sosomt.p for Sales Order Maintenance in MFG/PRO. The task is a type of transaction or event (for example, PostGL) that is typically a level lower than component. Use this field for reference when storing this kind of historical information is important. In particular, use it with OAGIS BODs where it maps to TASK in the SENDER segment of the CNTROLAREA.	N
@ACKLVLREQD <value>	The acknowledgment level the source application requested. Valid values are: • 0: No confirmation required • 1: Send message only if not successful • 2: Confirm on success or failure	N
@DOCLNG <value>	The primary language of the text fields in the document, as defined by the ISO language codes; for example, US for US English (maximum 2 characters). Use this field for reference when storing this kind of historical information is important. In particular, use it with OAGIS BODs where it maps to LANGUAGE in the SENDER segment of the CNTROLAREA.	N

Table B.1 — *Text File Control-Tag Format* — (Page 2 of 4)

Tag	Purpose	Req
@CODEPG <value>	The code page or character set (such as iso8859-1, ibm850) used to present, or format, the document data. If left blank, Q/LinQ assumes that any import document of this type uses the same code page as MFG/PRO.	N
@DATEFORM <value>	Describes the date format used to represent calendar dates in the document, using the values supported by Progress: mdy, dmy, ymd.	n
@NUMFORM <value>	Describes the format used to represent decimal numbers in the document, using the values supported by Progress. Possible values are: • American, where the comma separates thousands and the period is a decimal point • European, where the period separates thousands and the comma is a decimal point	N
@SRCUSERID <value>	The user ID of the person or process in the registered application initiating the document export or import if this value is fixed rather than variable. Use this field for reference when storing this kind of historical information is important. In particular, use it with OAGIS BODs where it maps to AUTHID in the SENDER segment of the CNTROLAREA.	N
@SRCPROC <value>	The source procedure that created or sent the document.	N
@MFGPROSITE <value>	The primary MFG/PRO host site this Q/LinQ installation is associated with. Use when Q/LinQ is deployed with multisite MFG/PRO implementations and the site code on MFG/PRO documents is populated.	N
@MFGPROKEY <value>	The document identifier containing key reference information (such as order-line number or item number) relevant to MFG/PRO application processing.	N
@EXTDOCKEY <value>	Any value significant to the external application that is useful for audit-trail purposes, such as a cross-reference to an earlier document imported from the application.	N
@DATA	Required. This line has no control-tag value. The transaction data follows this tag.	Y

Table B.1 — *Text File Control-Tag Format* — (Page 3 of 4)

Tag	Purpose	Req
@ORIGDOCID <value>	Q/LinQ-assigned document ID of an earlier document this new document is related to. Required only on acknowledgments, where it refers back to the Q/LinQ export document being acknowledged. Contrast with EXTDOCKEY, which is not a Q/LinQ-assigned identifier.	N
@CTRLTOT1 <integer>	User-defined control counters or totals related to the document. Not currently used by Q/LinQ.	N
@CTRLTOT2 <decimal>	User-defined control counters or totals related to the document. Not currently used by Q/LinQ.	N
@CTRLTOT3 <decimal>	User-defined control counters or totals related to the document. Not currently used by Q/LinQ.	N
@DATECREATE <value>	Date on which the document was originally created outside of Q/LinQ.	N
@TIMECREATE <value>	Time at which the document was originally created outside of Q/LinQ.	N
@TIMEZONE <value>	Time zone in which the document was originally created outside of Q/LinQ.	N
@@DOCEND	Required. Signals Q/LinQ that this is the end of the data. This line has no control-tag value.	Y
@@GROUPEND	Indicates the last of the documents belonging to a series or batch group.	N

Table B.1 — *Text File Control-Tag Format* — (Page 4 of 4)

ADI Tags

You can use the CIM tags, or keywords, of the application data interface (ADI) listed in Table B.2, without alteration, to import text files.

Note The ADI is associated with programs written in the Object-Based Component Model (OBCM). These programs no longer exist in MFG/PRO eB2.

Table B.2 ADI Tags of the CIM Interface

```
@@BATCHLOAD <procedure name>
@LOADPROC <load program>
@ERRORDATAFILE <error-out-file>
@SUCCESSDATAFILE <>
@DEFAULTSFILE <>
@ACTION <edit or delete>
<data lines>
.
.
.
@PROCESS
@@END
```

Application Control Tags

When the optional tags in Table B.3 are included with data imported from an application, they identify the import document so that Q/LinQ can look up its Import Specification Maint record. The tags must be before the data. The @DATA tag immediately precedes the data.

Table B.3 Application Control Tags

```
@APPID <value>
@DOCSTD <value>
@DOCTYPE <value>
@DOCREV <value>
@TRADPTRID <value>
@DATA <value>
```

Appendix C

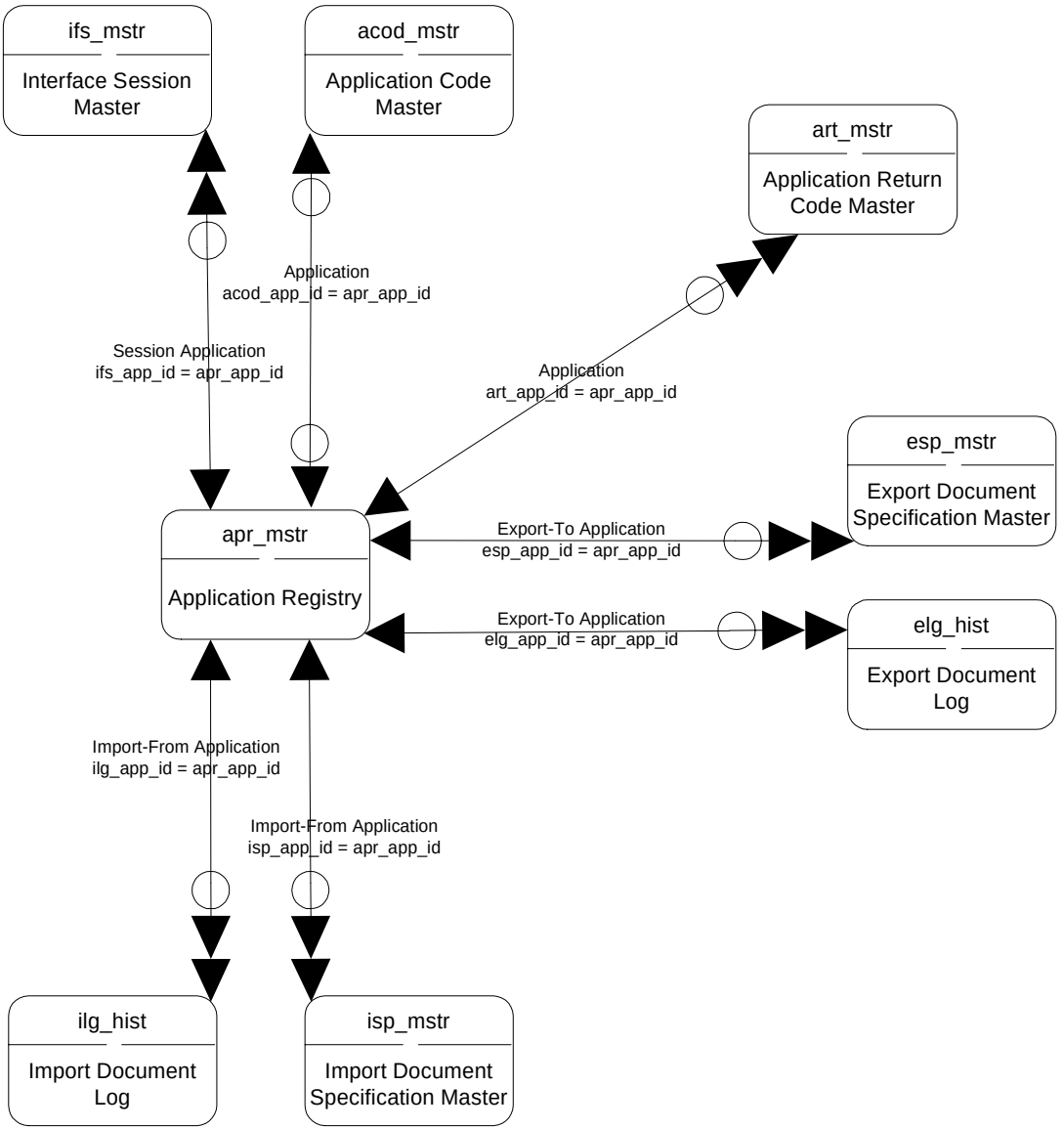
Entity Diagrams

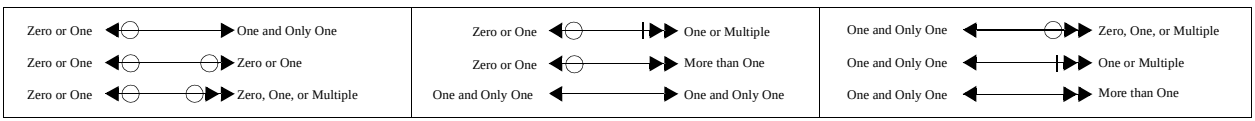
These diagrams are meant to be used in conjunction with database definitions. Refer to the *Database Definitions* technical reference for your version of MFG/PRO.

The entity diagrams are in alphabetical order starting on the next page.

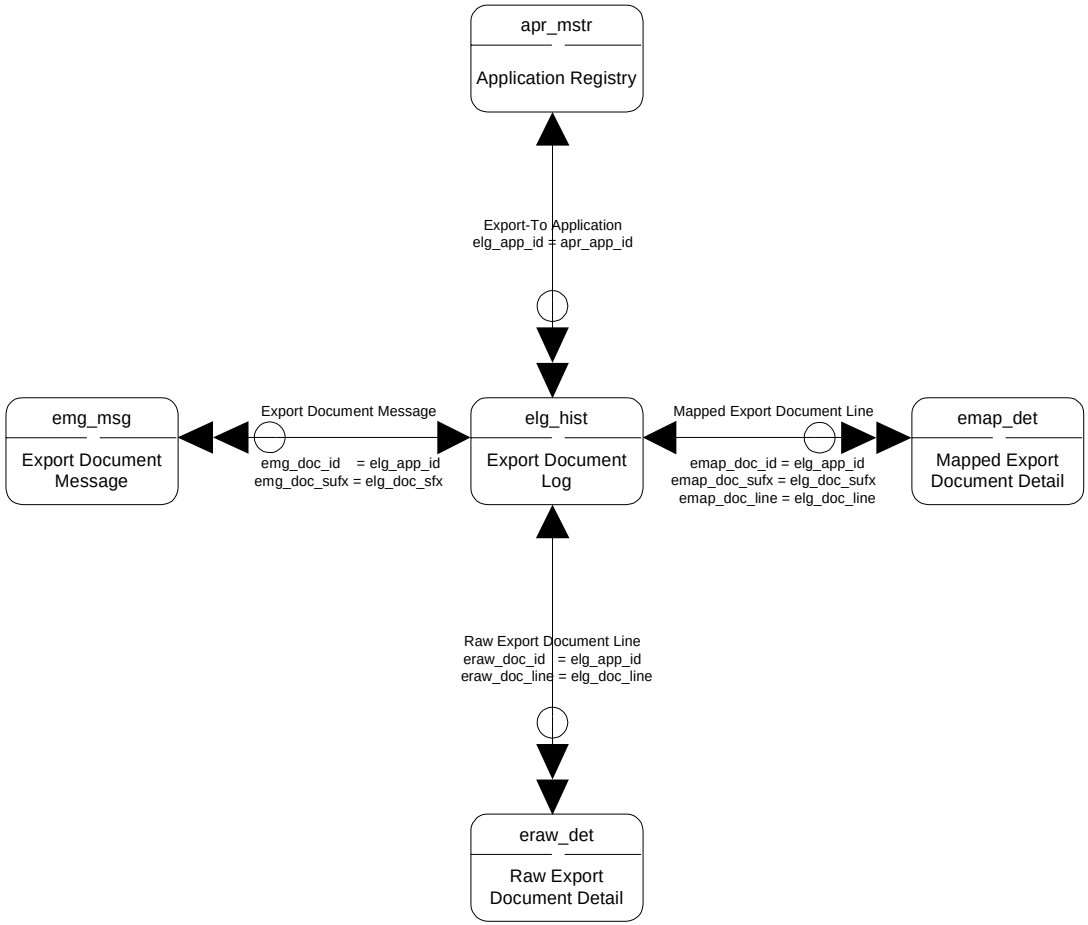
Q/LinQ I

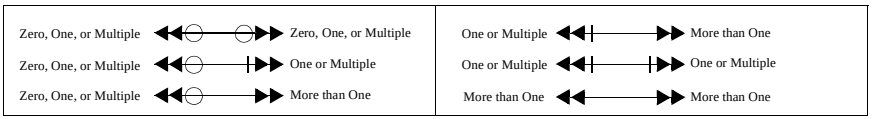
The acod_mstr table is not available in MFG/PRO 9.0.



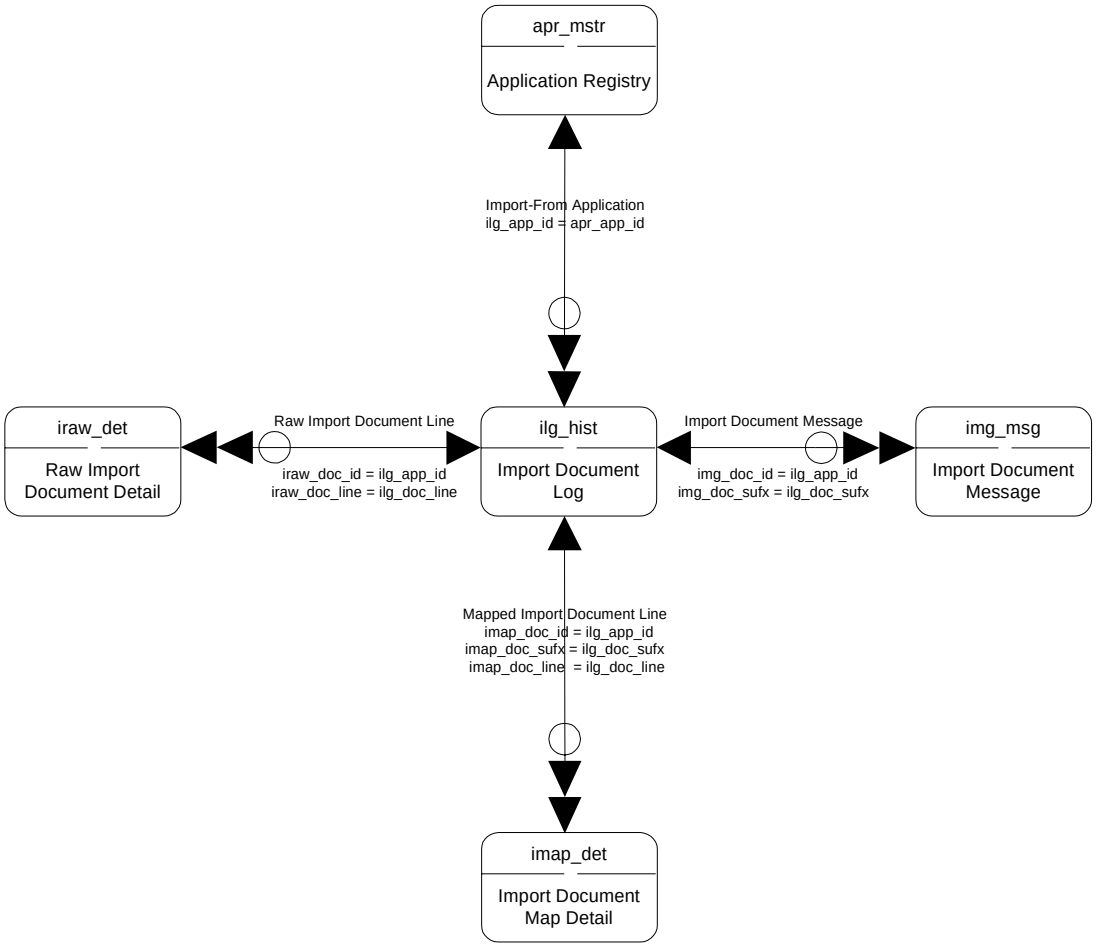


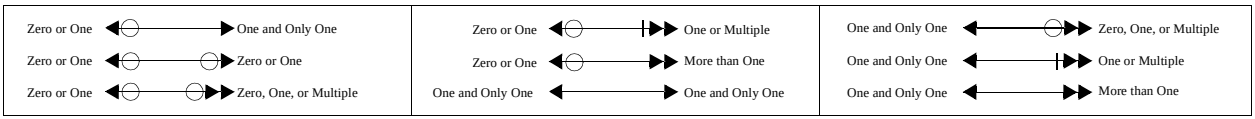
Q/LinQ II


apr_mstr
Application Registry
Export-To Application
elg_app_id = apr_app_id
emg_msg
Export Document Message
Export Document Message
emg_doc_id = elg_app_id
emg_doc_sufx = elg_doc_sfx
elg_hist
Export Document Log
Mapped Export Document Line
emap_doc_id = elg_app_id
emap_doc_sufx = elg_doc_sufx
emap_doc_line = elg_doc_line
emap_det
Mapped Export Document Detail
Raw Export Document Line
eraw_doc_id = elg_app_id
eraw_doc_line = elg_doc_line
eraw_det
Raw Export Document Detail


Zero, One, or Multiple
Zero, One, or Multiple
Zero, One, or Multiple
One or Multiple
Zero, One, or Multiple
More than One
One or Multiple
More than One
One or Multiple
One or Multiple
More than One
More than One

Q/LinQ III


apr_mstr
Application Registry
Import-From Application
ilg_app_id = apr_app_id
iraw_det
Raw Import Document Detail
Raw Import Document Line
iraw_doc_id = ilg_app_id
iraw_doc_line = ilg_doc_line
ilg_hist
Import Document Log
Import Document Message
img_doc_id = ilg_app_id
img_doc_sufx = ilg_doc_sufx
img_msg
Import Document Message
Mapped Import Document Line
imap_doc_id = ilg_app_id
imap_doc_sufx = ilg_doc_sufx
imap_doc_line = ilg_doc_line
imap_det
Import Document Map Detail


Zero or One
One and Only One
Zero or One
Zero or One
Zero or One
Zero, One, or Multiple
Zero or One
One or Multiple
Zero or One
More than One
One and Only One
One and Only One
One and Only One
Zero, One, or Multiple
One and Only One
One or Multiple
One and Only One
More than One

Glossary

Adapter. A program that mediates differences in data communications or remote method-invocation protocols. An adapter translates call signatures or parameters of one program into a form usable by another—the adapted program. This lets the two programs exchange data or lets the calling program invoke the services of the adapted program without having to modify code. Do not confuse with a data-mapper program, which mediates differences in application data content.

ADI. See *Application Data Interface (ADI)*.

ANSI X12. American National Standards Institute, Standard X12.

Apache Software Foundation. A membership-based, not-for-profit corporation that exists to provide organizational, legal, and financial support for the Apache open-source software projects.

API. See *Application Program Interface (API)*.

Application Data Interface (ADI). A data-transfer capability in MFG/PRO programs using the Object-Based Component Model (OBCM) architecture. OBCM allows data streams going into or out of MFG/PRO to be redirected from or to an external text file. Unlike CIM Interface records, the data format of ADI records is independent of the user interface presentation. It includes metadata consisting of Progress-qualified field names as well as data values, providing improved flexibility and maintainability.

Application Program Interface (API). A set of entry points into and calls out of an application. External applications use these to link to and pass data or provide services to an application without having to modify its internal processing. Commercial APIs are typically exposed to external programs through designated program libraries with the signature of each call, the call parameters, and the procedures for use well documented.

Asynchronous Processing. Processing that an external event (for example, committing a transaction to the database) initiates but is not completed as part of the same unit of work that initiated it. Asynchronous processing can complete long after the originating event completes. The originating application is not blocked from further processing before the asynchronous processing finishes.

Batch of One Immediate (BOOI). The preparation of transactions for immediate transport between applications as an application event occurs, as opposed to subsequent or periodic extraction and batching of business data for later transport. BOOI data is not necessarily synchronously received or processed at the destination. Implementations

often employ a store-and-forward architecture to first queue the data, then send it later based on data communications availability, cost, and timing requirements at the destination.

BOD. See *Business Object Document (BOD)*.

BOOI. See *Batch of One Immediate (BOOI)*.

Business Object Document (BOD). The OAGIS implementation of the concept of document (see *Document*). Business object documents specify the business data that are exchanged between different kinds of applications.

Callback Function. An API entry point that permits external programs to invoke services of an application through a direct program call into that application. References to such entry points are often passed to the external applications through subscribe or connect functions.

Common Object Request Broker Architecture (CORBA). An architecture and specification for creating, distributing, and managing distributed program objects in a network. Programs at different locations and developed by different vendors communicate in a network through an interface broker. See also *Object Request Broker (ORB)*.

Document. General term used to signify an instance of a business transaction, such as a sales order or invoice. In Q/LinQ, a document is an electronic representation of business data sent to or received from an external application or trading partner. This term is generally used at the application level of abstraction, in preference to the somewhat more technical terms transaction or message. See also *Mapped Document* and *Raw Document*.

EDIFACT. Electronic Data Interchange for Administration, Commerce, and Transport. Generally used to refer to the set of international standards that were developed to enable EDI among businesses located in North America, Europe, and other geographic regions. In 1987, the EDIFACT syntax proposals were accepted by the ISO Technical Board and the number ISO 9735 was allotted. These standards are application, communications medium, and machine-independent.

Electronic Data Interchange (EDI). The electronic transport of business documents (such as sales, purchases, shipment notices, delivery schedules) between trading partners. Various industries and QAD target markets, such as automotive manufacturing and consumer-products distribution, use EDI extensively. These industries generally implement EDI using commercial enabling software products that support data transport, often using value-added-network (VAN) services, and data translation to or from the industry-defined EDI standard formats.

Extensible Markup Language (XML). A flexible way to create information formats and share both the format and the data on the Web, intranets, and elsewhere. XML is similar to HTML; both contain markup symbols to describe the contents of a page or file. HTML, however, describes only how a Web page displays. XML describes its content and data types.

Extensible Stylesheet Language (XSL). A language for formatting an XML document; for example, showing how the data described in the XML document should be presented in a Web page. XSLT shows how the XML document should be reorganized into another data structure (which could then be presented by following an XSL style sheet).

Extensible Stylesheet Language Transformation (XSLT). A standard way to describe how to transform the structure of an XML document into an XML document with a different structure. The coding for the XSLT is also referred to as a style sheet and can be combined with an XSL style sheet or be used independently.

External System. A set of programs running on a computer, operating system, or computer network that is different from the MFG/PRO installation that Q/LinQ is running in. The external system could be another MFG/PRO installation.

Forwarding. the action taken when an API exports documents from one MFG/PRO module and places them in the import queue for processing by another MFG/PRO module within the same system.

Gateway. An MFG/PRO load or update program that either extracts data for sending to an external application or processes messages received from an external application. Gateways are responsible for most or all of the direct MFG/PRO database access required to implement APIs. They are internal to MFG/PRO, not exposed as part of an API. Unlike programs invoked using the Q/LinQ user-interface emulation mode, gateway programs have no user interface and are generally better suited for reliable and faster processing of complex, variable-input data.

Guaranteed Delivery. Capability supported by various message-oriented middleware and EDI packages that assures a message sent from a source node and queue is delivered to its destination node and queue. It guarantees only delivery, not that the delivered message is received or processed by the destination application.

Guaranteed Receipt. Capability supported by various message-oriented middleware and EDI packages that assures that a message sent from a source application is received by the destination. It guarantees only receipt, not that the received message is successfully processed at the destination. Synonymous with the functional acknowledgment often implemented in EDI products.

Handle. A temporary identifier assigned to an object during a communication sessions.

Interface Message. Also known as interface document. A single package of data transported between separate applications that has integrity and is complete for both applications. The applications exchanging interface messages are often but not necessarily distributed across multiple platforms and/or network locations.

Mapped Document. A document whose format has been transformed to a format the destination application can read. See also *Document*.

Mapper. A program or a specification file used by a commercial mapping program to transform application data from native MFG/PRO format into an external format or vice versa.

Message-Oriented Middleware (MOM). A class of middleware that provides data- or message-transport services between applications when the source and destination applications are typically distributed on different platforms and development or runtime environments. Some MOM packages also include data transformation or mapping services.

Middleware. General term for off-the-shelf connectivity software that supports distributed processing at run time and application development environments. Middleware

operates between the application logic and the underlying physical network. It typically handles general functions that are required to implement connectors with external applications, such as reliable message transport, message queuing, database access, distributed transaction processing, and data transformation.

MOM. See *Message-Oriented Middleware (MOM)*.

OAG. See *Open Applications Group (OAG)*.

OAGIS. See *Open Applications Group Integration Specification (OAGIS)*.

Object Management Group (OMG). A nonprofit organization dedicated to creating an interoperability standard for independently developed applications across heterogeneous networks of computers. The most significant specification released by OMG is CORBA. See *Object Request Broker (ORB)* and *Common Object Request Broker Architecture (CORBA)*.

Object Request Broker (ORB). A class of middleware that enables objects to request and provide services from or to each other over networks with different communications protocols. The objects can reside on different hardware platforms and applications. There are several emerging standards for ORB communication, chief among them Common Object Request Broker Architecture (CORBA) by the Object Management Group (OMG), and Distributed Component Object Model (DCOM) by Microsoft.

OLE/COM. Object linking and embedding/ Component Object Model. Microsoft's framework for a compound document technology. Briefly, a compound document is something like a display desktop that can contain visual and information objects of all kinds: text, calendars, animations, sound, motion video, 3-D, continually updated news, controls, and so forth. Each desktop object is an independent program entity that can interact with a user and also communicate with other objects on the desktop. Part of Microsoft's ActiveX technologies, OLE takes advantage and is part of a larger, more general concept, the Component Object Model (COM) and its distributed version, DCOM). An OLE object is necessarily also a component (or COM object).

OMG. See *Object Management Group (OMG)*.

Open Applications Group (OAG). A nonprofit organization composed primarily of ERP software companies established to further multi-vendor integration by establishing and publishing specifications for the exchange of business data across applications.

Open Applications Group Integration Specification (OAGIS). The primary body of standards established and published by the OAG.

ORB. See *Object Request Broker (ORB)*.

Process Acknowledgment. Also know as process confirmation**.** Ability of one application to notify another that a previously sent message has been successfully processed. Process acknowledgment is stronger than the functional acknowledgment employed by some EDI standards in that it implies the message has not only been received but also processed by the target application. It is implemented in OAGIS through the BOD `CONFIRM_BOD`.

Protocol. The rules, formats, and functions components that a network or communication system use.

Publish-Subscribe. An asynchronous exchange of messages between applications or objects in which various parties request to receive particular types of documents or event notifications (subscribe) as other parties make them available (publish). Neither the

publishing nor subscribing parties wait for a reply. Publish-subscribe usually requires middleware such as an ORB or MOM to queue the published messages and to broker communication between publishers and subscribers. Often contrasted with *Request-Reply*.

QOS. See *Quality of Service (QOS)*.

Quality of Service (QOS). A network methodology that attempts to measure, improve, and—to some extent—guarantee in advance transmission rates, error rates, and other characteristics. QOS is of special importance for continuous data transmission.

Raw Document. A document that has not been altered between the source application and destination application. See also *Document*.

Registry. A set of tables in Q/LinQ containing relatively static information about the external applications that Q/LinQ can connect with, the maps and adapters used to connect with them, and the document or message types supported for each application. Maintain the registry entries using Register External Application, Export Specification Maint, and Import Specification Maint.

Request-Reply. A synchronous exchange of messages between applications or objects in which the initiator or client makes a service request and blocks further execution until it receives a reply from the server. Request-reply is often implemented using remote procedure call (RPC) protocols or through ORBs. Often contrasted with *Publish-Subscribe*.

Session. In Q/LinQ, a single procedure that is publishing, sending, receiving, processing, archiving, or deleting documents. A Q/LinQ session can import or export one document or a group of documents in a single file or from an executing process.

Socket. A convention for connecting with and exchanging data between two program processes within the same computer or across a network. A socket represents the end point in a network connection. Sockets are created and used with a set of programming requests or function calls sometimes referred to as the socket application program interface (API). The most common sockets API is the Berkeley UNIX C language interface.

Store and Forward. An approach for passing data between applications in which each message is queued for sending by the source application and then transported asynchronously to its destination. The source application does not wait for the message to be delivered to or processed by the destination.

Synchronous Processing. Processing that is completed as part of the same application event and unit of work that initiated it. Typically, a reply or result is required from the synchronous process before execution can continue in the initiating application. Because synchronous processes constitute the critical path of a transaction, they tend to affect application performance more than asynchronous ones.

Universal Unique Identifier (UUID). A hexadecimal number including a time stamp and a host identifier. Applications use UUIDs to identify many kinds of entities.

Winsock. A programming interface and the supporting program that handles input/output requests for Internet applications in a Windows operating system. Winsock is an adaptation for Windows of the Berkeley UNIX sockets interface

XML. See *Extensible Markup Language (XML)*.

XSL. See *Extensible Stylesheet Language (XSL)*.

XSLT. See *Extensible Stylesheet Language Transformation (XSLT)*.

Index

Numerics

36.3.18 106
36.5.18 106
36.8.1 23, 27, 43, 48, 89, 95, 96, 107, 127, 130, 138, 139
36.8.3 127
36.8.1.1 23, 27, 43, 48, 89, 95, 96, 107, 130, 138, 139
36.8.1.2 89, 96, 109, 115, 116
36.8.1.3 89, 90, 95, 96, 109, 115, 116, 139
36.8.1.20 116
36.8.2 89, 96, 109, 115, 116
36.8.3 89, 90, 95, 96, 109, 115, 116, 139
36.8.5 118, 131, 139, 145
36.8.7 67, 98, 119, 122, 129, 131, 132
36.8.9 67, 119, 122, 139
36.8.10 67, 98, 142
36.8.11 146
36.8.12 89
36.8.13 68, 147
36.8.14 148
36.8.15 149
36.8.16 127
36.8.17 151
36.8.18 150
36.8.23 65, 67, 153
36.8.24 89, 104, 131
36.15 142

A

adapter programs
  defined 76
  message-oriented middleware (MOM) 119
  MOM communication 23, 43
  non-MOM communication 22, 42
  `QqMomAdapter` 101
  register 138
  restarting after Q/LinQ failure 119
Apache Software Project 29
application data interface (ADI) 95, 173
application ID 134
application program interfaces (APIs)
  communication APIs 99
  communication testing 122
  mapping APIs 96
  processing APIs 93
  publishing APIs 92
application queues, create 26, 46
archive directory 65
archive/delete Q/LinQ data 153
asynchronous processing 82
autoacknowledgment 87
  CONFIRM_BOD 88
  e-mail notification 90
  level 89
  processing 89

B

batch group. *See* document groups
batch of one immediate (BOOI) 82
batch session, import processing 144
bin directory 65
business object documents (BODs)
  OAGIS standards 82
  used by Q/LinQ 157
  XML 82

C

channel initiator 47
channel, create 26, 45
CIM interface
  debugging 146
  Q/LinQ similar to 94
  Q/LinQ use of 142
classes directory 65
`CLASSPATH` 27, 48
Code Mapping Maintenance 116

com.ibm 27, 48
common object request broker architecture (CORBA) 91
communication
  middleware setup 23, 43
  non-middleware setup 22, 42
  testing APIs 122
communication API 99
communication testing 122
CONFIRM_BOD 88, 90
control file 104
CORBA 91

D
data directory 66
data exchange 81
database
  direct access, register for 95
DCOM 91
Debug CIM Document 146
delete/archive Q/LinQ data 153
delimiter, triplet 109
distributed component object model (DCOM) 91
Document Control Tag Maint 149
document exchange
  text files 90
  with APIs 91
document groups 84
  batch 84
  individual 85
  mapping 98
  processing 143
  series 84
  transmission failure 128
  types 84
document standards 82, 115, 135
documents. *See* Q/LinQ documents
Dump Export/Import Doc for Edit 68
Dump Export/Import Docs
  Edit, for 147
  File, to 150

E
e-mail
  Q/LinQ notification 90
  Q/LinQ use of 106
errata, installation 8
explog directory 66
Export Specification Maint 89, 96
  document specifications 109, 115
  trading partners, MOM channels and queues 116
Export/Import Doc Delete/Archive 65, 67, 153
Export/Import Document Query 127, 142
Export/Import Document Report 142, 151
exporting
  continuously 131, 134
  document specifications 109, 115
  documents 129
  multiple sessions 132
  processing stages 128
  register external application 130
Extended Stylesheet Language Transformation (XSLT)
  See XSLT stylesheet
extensible markup language (XML) 91
external applications
  exporting, register 130
  importing, register application 139
  register 107
  register multiple with MOM 112
  register multiple without MOM 112

F
forwarding Q/LinQ documents 93, 131

G
gateway programs
  Q/LinQ defined 77
  Q/LinQ use of 95
groups, document. *See* document groups

I
implog directory 66
Import Specification Maint
  autoacknowledgment processing 90
  database access, direct 95
  document specifications 109, 115
  mapping documents 127
  mapping procedure, default 96
  receiving documents 139
  trading partners, MOM channels and queues 116
importing
  continuously 137
  document specifications 109, 115
  documents 136
  multiple sessions 132
  processing stages 128
  register external application 139
individual-document group. *See* document groups
initialization files
  qqapi.ini 19, 39, 70
  QqMomAdapter.ini 23, 43
installation
  directories 64
    archive 65

bin 65
classes 65
data 66
explog 66
implog 66
lib 68
mapspec 67
mom 67
samples 68
scripts 67
errata 8
Java requirements 9
MQSeries 24, 44
MQSeries requirements 9
programs, Windows 34
scripts, UNIX 17
server requirements 9
UNIX 15
UNIX CD mount commands 16
Windows client 37
Windows file server 34
Interface Session Monitor 118, 145
document ID 85
export sessions 131
import sessions 139
processing session 143
session ID 85

J

Java Client 25, 45
Java Virtual Machine 25, 45

L

lib directory 68
listener, start 26, 46
log files, MFG/UTIL 11

M

mapping
applications 77
export documents 99
import documents 98
log table 98
one-to-many 98
one-to-one 98
procedure, register 96
registration required 127
triplet delimiter 109
mapping API 96
mapspec directory 67
menu in MFG/PRO 79
menu reference conventions 4
Mercator
invoking 98
MFG/UTIL
keyboard commands 10
log files 11
middleware
channels and trading partners 116
defined 77
message-oriented (MOM) 83, 101
MQSeries 101
object request broker (ORB) 83
queues and trading partners 116
register multiple applications with 112
register multiple applications without 112
set up communication 23, 43
MOM adapter
`QqMomAdapter` 101
starting 119
stopping 120
mom adapter
directory for 67
MOM. *See* middleware
MQSeries
application queues 26, 46
channel 26, 45
channel initiator 47
client libraries 27, 48
host and port 28, 49
install and configure 24, 44
installation requirements 9
listener, start 26, 46
preventing data loss 121
Q/LinQ supported MOM 101
queue manager 25, 45
server 25, 45

N

network ports
allocate and specify 22, 26, 42, 47
MQSeries 28, 49
non-Progress programs
UNIX installation 15

O

OAGIS 91
document standard 82
importing BODs 139
standards 82
XML BODs 82
object request broker (ORB) 83
Open Applications Group (OAG) 82
Open Applications Group Integration Specification.

See OAGIS

P

ports. *See* network ports
Process Import Documents 98
  batch session 144
  command window, repositioning 144
  continuously 143
  installation directory 67
  map documents 142
  Mercator, invoking 98
  source application ID 142
  update MFG/PRO database 142
Process Received Acknowledgment 89
processing API 93
processing stages 126
Progress
  requirements 9
PROPATH 19, 39
publishing API 92

Q

Q/LinQ
  autoacknowledgment 87
  communication API 99
  interface message 82
  introduction to 74
  log for mapping activity 98
  mapping API 96
  menu 79
  processing API 93
  publishing API 92
  restarting after failure 119
  session ID 85
  setting up 103–124
  starting and stopping 118
  using 125–154
Q/LinQ Control
  autoacknowledgment level 89
  setting values in 104
  system ID field 131
Q/LinQ documents
  defined 82
  deleting and archiving 153
  document ID 85
  edit raw data 147
  editing imported 147
  exchanging 90, 91
  forwarding 93, 131
  frequency of deletion 153
  messages 82
  processing import 142
  sequence 100
  standards for 82
`qqapi.ini` 19, 39, 70
`qqArchiveDir` 18, 36
`qqEditDir` 18, 36
`qqExpLog` 18, 37
`qqImpLog` 18, 37
`QqMomAdapter` 101
  configure Q/LinQ client 27, 48
  initialization file 23, 43
  reserve ports 26, 47
qqrecv 122
QqRecv.class 122
qqsend 122
QqSend.class 122
`qqWorkDir` 18, 37
quality of service (QOS) 110
queue manager 25, 45

R

Receive Import Documents 67, 139
  restarting Q/LinQ after failure 119
  testing communication 122
Register External Application
  adapter programs, register 138
  autoacknowledgment level 89
  database access, direct 95
  Default Communications Parameters 109
  Default Data Mapping Parameters 109
  exporting documents 130
  importing documents 139
  Interface Control Parameters 108
  mapping documents 127
  mapping procedure, default 96
  Q/LinQ-adapter ports, specify 23, 27, 43, 48
  registering external applications 107
  Return Codes and Messages 112
  Transaction Processing Programs 111
Reload Edited Export/Import Doc 148
requirements, installation
  Java 9
  MQSeries 9
  server 9

S

samples directory 68
scripts
  UNIX installation 17
scripts subdirectory 67
Send Export Documents 131
  exporting documents 129
  forwarding documents to MFG/PRO 131

installation directory 67
Mercator, invoking 98
restarting after Q/LinQ failure 119
testing communication 122
using 132
series group. *See* document groups
session (Q/LinQ) 84

T

testing communication 122
time zones 105
trading partners
MOM channels and queues 116
sending documents to 136
triplet delimiter 109

U

UNIX
CD mount commands 16
installation 15
user count 100
User Maintenance 106
user-interface emulation 94

V

version file 64

W

Windows
client installation 37
file server installation 34

X

XML 82, 91, 158
XSLT stylesheet
CONFIRM_BOD 158
XSLT stylesheets 97
debugging in Windows 51
debugging on UNIX 30
setting up on UNIX 29
setting up on Windows 50