

**Tivoli Data Exchange  
Version 1.2.5**

**User's Guide**

Copyright 2001, 2002 Tivoli, Inc. All rights reserved.

5/20/02

Tivoli Data Exchange, Tivoli, and the Tivoli logo are registered trademarks or trademarks of Tivoli, Inc. AS/400, AIX, MQSeries, and OS/390 are registered trademarks of IBM Corp. InstallShield is a registered trademark of InstallShield Software Corporation. Win 32 is a registered trademark of Microsoft Corp. UNIX is a registered trademark of The Open Group. Solaris and Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. All other trademarks are property of their respective owners.

# Contents

<b>chapter 1</b>	<b>About This Guide . . . . .</b>	<b>13</b>
	About Tivoli Data Exchange . . . . .	13
	Who Should Read This Guide . . . . .	15
	How This Guide Is Organized . . . . .	16
	Contacting Customer Support . . . . .	17
	Where to Look for More Information . . . . .	17
	Conventions . . . . .	18
<b>chapter 2</b>	<b>Tivoli Data Exchange Concepts . . . . .</b>	<b>19</b>
	Introduction . . . . .	19
	Why FTF? . . . . .	19
	Architectural Overview . . . . .	21
	FTF Interfaces . . . . .	22
	FTF Manager . . . . .	23
	FTF Sender . . . . .	24
	FTF Receiver . . . . .	24
	FTF Status . . . . .	24
<b>chapter 3</b>	<b>Tivoli Data Exchange Use and Customization . . . . .</b>	<b>27</b>
	FTF Components . . . . .	28
	The FTF Manager . . . . .	28
	The FTF Sender . . . . .	31
	The FTF Receiver . . . . .	31
	Starting FTF . . . . .	32
	Starting FTF on OS/390 Systems . . . . .	33
	Starting FTF on Win 32 . . . . .	34
	Starting FTF on AS/400 . . . . .	36
	Starting FTF in UNIX . . . . .	38
	Starting Components Individually . . . . .	38
	Getting Status Information from FTF Components . . . . .	42
	The FTF Manager and the Status Subsystem . . . . .	43
	FTF Sender and the Status Subsystem . . . . .	44
	FTF Receiver and the Status Subsystem . . . . .	45

Shutting Down FTF Components .....	46
Using the FTFEND Command .....	46
Using FTFSTART Address Space Commands .....	48
Security .....	50
OAM Security on OS/390 .....	50
Using the Options File .....	53
Typical Data-Transfer Scenarios .....	54
Scenario One .....	55
Scenario Two .....	56

## **chapter 4      Status Offload Daemon. . . . . 57**

Overview .....	58
Starting and Stopping the Status Offload Component .....	60
Starting the FTFStatD Component from the Win 32 Service .....	60
Stopping the FTFSTATD Component from the Win 32 Service .....	60
Configuration File and Command-Line Definitions .....	61
Defining the Configuration File for Status Offload .....	61
Defining the Command Line for Status Offload .....	63
Defining the Command Line for Retrieval from the Sample Relational Database .....	65
Recommended Database Schema .....	65

## **chapter 5      Tivoli Data Exchange User Exits . . . . . 67**

Exploiting User Exit Facilities in FTF .....	68
FTF User Exit Overview .....	70
Manager Pre-Process Exit .....	71
Manager Post-Process Exit .....	73
Sender Pre-Process Exit .....	76
Sender Post-Process Exit .....	78
Receiver Pre-Process Exit .....	80
Receiver Post-Process Exit .....	82
Reporting Status Information from Exit Modules .....	84
Customizing and Implementing FTF Exits .....	85
Sample Exits .....	86
Compiling and Generating DLLs .....	87
Making Exits Available .....	90

## **chapter 6      Security Authorization Exits . . . . . 93**

Customizing the Authorization Module .....	94
Using Authorization Module Data Structures .....	94
Authorization Module Return Codes .....	95

Installing the Security Authorization Exit . . . . .	96
Using FTFSTART on OS/390 . . . . .	97
Restarting the TDE Service . . . . .	98
Restarting Components Individually . . . . .	99
Interface for Customization . . . . .	100
Security Authorization Exit Reference . . . . .	102

## **chapter 7      Working with AS/400 Files . . . . . 105**

Overview . . . . .	106
Modifying the FTF Configuration File . . . . .	107
Using the FTF Command . . . . .	108
Sending AS/400 Physical or Logical File Data . . . . .	109
Examples . . . . .	111
Sending to a Fixed-Length Text File . . . . .	112
Sending to a Delimited Text File . . . . .	113
Sending from a Fixed-Length Text File . . . . .	115
Sending from a Delimited Text File . . . . .	116
Sending AS/400 Physical Files . . . . .	117
Sending AS/400 Logical Files . . . . .	119
Determining the Transfer's Success . . . . .	121
AS/400 Sender Status Errors . . . . .	122
AS/400 Receiver Status Errors . . . . .	123

## **chapter 8      Using the ISPF User Interface . . . . . 127**

ISPF Panel Structure . . . . .	128
Accessing the ISPF Panels . . . . .	129
Requesting a Data Transfer . . . . .	129
Entering Transfer Options . . . . .	132
Entering Application Integration Options . . . . .	138
Setting Stage Options . . . . .	141
Setting Reply/Notify Options . . . . .	144
Setting MVS/ESA (OS/390) Allocation Options . . . . .	145
Setting AS/400 File Options . . . . .	148
Setting Connector Software Options . . . . .	149
Staging Data-Transfer Requests . . . . .	153
Viewing Status Information . . . . .	155
Log Information Display . . . . .	159
Summary Information Displays . . . . .	159
Cancelling a Data-Transfer Request . . . . .	160
Pinging Components . . . . .	161
Shutting Down Components . . . . .	164
Exiting the ISPF Panels . . . . .	165

## chapter 9      **Using the 5250 User Interface . . . . . 167**

Accessing the 5250 Interface Commands . . . . .	168
FTF Request . . . . .	172
FTF Request Parameters – Group 1 . . . . .	172
FTF Request Parameters – Group 2 . . . . .	174
FTF Request Parameters – Group 3 . . . . .	176
FTF Request Parameters – Group 4 . . . . .	179
FTF Request Parameters – Group 5 . . . . .	184
FTF Request Parameters – Group 6 . . . . .	186
FTF Request Parameters – Group 7 . . . . .	188
FTF Request Parameters – Group 8 . . . . .	189
FTF Request Parameters – Group 9 . . . . .	190
FTF Request Parameters – Group 10 . . . . .	191
FTF Request Parameters – Group 11 . . . . .	192
FTF Request Parameters – Group 12 . . . . .	193
FTF Ping . . . . .	194
FTF Ping Parameters – Group 2 . . . . .	195
FTF Cancel . . . . .	196
Cancel FTF Request – Group 1 . . . . .	197
FTF Cancel Parameters – Group 2 . . . . .	198
FTF Status . . . . .	199
FTF Status Parameters – Group 1 . . . . .	199
FTF Status – Group 2 . . . . .	201
FTF Status Parameters – Group 3 . . . . .	202
FTF End . . . . .	203
FTF Stage . . . . .	206
FTF Stage Parameters – Group 1 . . . . .	206
FTF Stage – Group 2 . . . . .	208
FTF Configuration . . . . .	209
FTF Config – Group 1 . . . . .	209
FTF Config – Group 2 . . . . .	210
Starting an FTF Manager . . . . .	211
Start FTF Manager – Group 2 . . . . .	213
Starting an FTF Sender . . . . .	214
Start FTF Sender – Group 2 . . . . .	215
Start FTF Sender – Group 3 . . . . .	216
Starting an FTF Receiver . . . . .	217
Start FTF Receiver – Group 2 . . . . .	219
Start FTF Receiver – Group 3 . . . . .	220
Starting the FTF Logger . . . . .	221
FTF Logger – Group 1 . . . . .	221
FTF Logger – Group 2 . . . . .	223
Logging off FTF . . . . .	224

## **chapter 10      Using the FTF GUI. . . . .227**

Assumptions . . . . .	228
Starting the Tivoli Data Exchange GUI . . . . .	228
Viewing Data-Transfer Status Information . . . . .	229
Filtering Status Information . . . . .	229
Viewing Status Information . . . . .	233
Viewing Detailed Status Information . . . . .	235
Deleting Status Records . . . . .	236
Monitoring Status Information . . . . .	237
Changing the Polling Interval . . . . .	237
Submitting a Data-Transfer Request . . . . .	238
Setting Source File Options . . . . .	241
Setting Target File Options . . . . .	243
Setting Job Options . . . . .	245
Setting User Options . . . . .	247
Setting User Exits . . . . .	248
Removing User Exits . . . . .	251
Pinging Tivoli Data Exchange Components . . . . .	251
Staging a Data-Transfer Request . . . . .	252
Canceling a Transfer Request . . . . .	255
Shutting Down Tivoli Data Exchange Components . . . . .	257
Stopping Monitoring . . . . .	258

## **chapter 11      Tivoli Data Exchange Explorer . . . . .259**

Assumptions . . . . .	260
Software Requirements . . . . .	260
Tivoli Data Exchange Explorer Overview . . . . .	261
Using the Tivoli Data Exchange Explorer . . . . .	262
Customizing Your Tivoli Data Exchange Explorer . . . . .	263
Getting Started with Tivoli Data Exchange Explorer . . . . .	265
Viewing Status Information Using Tivoli Data Exchange Explorer . . . . .	267
Viewing the Tivoli Data Exchange Component Log Queue . . . . .	270
Adding, Modifying, and Deleting Data Exit Definitions . . . . .	271
Creating, Modifying, and Deleting Tivoli Data Exchange Requests . . . . .	274
Making an FTFping Request from Tivoli Data Exchange Explorer . . . . .	287
Scheduler . . . . .	290
Scheduling a Repeating Task . . . . .	290
Scheduling a Onetime Task . . . . .	299
Deleting a Scheduled Task . . . . .	303
Restarting the Scheduler . . . . .	303

<b>chapter 12</b>	<b>Directory Monitor (FTFDIRMN). . . . .</b>	<b>305</b>
	Overview . . . . .	306
	Configuring FTFDIRMN . . . . .	306
	FTFDIRMN Rules File . . . . .	306
	FTFDIRMN Configuration File . . . . .	310
	FTFDIRMN Command-Line Arguments . . . . .	310
 <b>chapter 13</b>	 <b>Tivoli Data Exchange Client for 4690 . . . . .</b>	 <b>313</b>
	Overview . . . . .	314
	What Is an MQSeries Client? . . . . .	314
	Why Use MQSeries Clients? . . . . .	315
	System and Installation Requirements . . . . .	316
	Installing the Client for 4690 . . . . .	316
	Adding Logical Filenames . . . . .	317
	Specifying File Distribution Types on the Command Line . . . . .	317
	Specifying the Target File Distribution Type in the Configuration . . . . .	318
	Initializing the Client for 4690 . . . . .	318
	Tivoli Data Exchange Manager Startup . . . . .	318
	Tivoli Data Exchange Sender Startup . . . . .	319
	Tivoli Data Exchange Receiver Startup . . . . .	320
	Configuring the Client for 4690 . . . . .	321
	Setting Up the MQSeries Client . . . . .	321
	Setting Up Background Processes for Components . . . . .	323
	Starting Background Processes for Components . . . . .	325
	Setting Up Clients on Another Platform . . . . .	326
	Using Exits on the 4690 Client . . . . .	329
	Tivoli Data Exchange Commands . . . . .	329
	Overview . . . . .	329
	FTF . . . . .	330
	FTFMGR . . . . .	330
	FTFRCV . . . . .	330
	FTFSDR . . . . .	330
	FTFPING . . . . .	330
	FTFEND . . . . .	331
 <b>chapter 14</b>	 <b>Using Pools . . . . .</b>	 <b>333</b>
	About Pools . . . . .	334
	Increasing Throughput . . . . .	334
	Load Balancing . . . . .	334
	Overcoming Queue Capacity Limitations . . . . .	334
	Using Pools with FTF . . . . .	335



Designing the Pools	335
Case Study	335
Implementing the MQSeries Objects	338
Defining the Transmission Queues	339
Defining the Local Queues	339
Defining the Queue Manager Aliases	340
Defining the Channels	340
Defining Pools in the FTF Configuration File	341
Specifying a Pool	344
Command-Line Interface	345
C API	345
COBOL API	346
ISPF Interface	346
5250 Interface	346
FTF GUI Interface	346

**chapter 15    Setting Up Configuration Queues . . . . . 347**

Overview to Setting Up a Configuration Queue .....	348
Setting the FTF Configuration Queue Stanza .....	348
Loading the Configuration Queue .....	349
Using the Configuration Queue .....	350
Setting Environment Variables .....	351
Changing Configuration Queue Information .....	351
Multiple Instances of FTF .....	351

<b>chapter 16</b>	<b>Logging Tivoli Data Exchange . . . . .</b>	
	<b>Information . . . . .</b>	<b>353</b>

Tivoli Data Exchange Logging Options .....	354
Writing to a Log File .....	354
Removing Log Information from Standard Output .....	355
Log File Contents .....	356
Using the FTFLOG Process .....	356
Configuring a Log Queue .....	357
Retrieving Information from Log Queues .....	357
Stopping the FTFLOG Daemon .....	361

**Index . . . . . 363**



---

---

# About This Guide

The *Tivoli Data Exchange User's Guide* describes how to use *Tivoli Data Exchange*(TDE) to create enterprise application-to-application integration solutions.

This chapter contains guidelines about the information in this manual and the conventions used to present the information. It contains the following sections:

Section	Page
About Tivoli Data Exchange	13
Who Should Read This Guide	15
How This Guide Is Organized	16
Contacting Customer Support	17
Where to Look for More Information	17
Conventions	18

## About Tivoli Data Exchange

Tivoli Data Exchange enables the collection and distribution of business-critical data, regardless of content type or size, across major platforms and network protocols. The benefits include reduced transfer times, saved network bandwidth, and easy scalability to meet growing e-business needs. Tivoli Data Exchange leverages and exploits the inherent strengths of MQSeries to provide secure, efficient, and reliable transfer of data across all platforms in an enterprise through compression and parallel delivery paths. It uses an IBM MQSeries engine for any-to-any connection, assured delivery, and data integrity checking. Tivoli Data Exchange caters to heterogeneous networks by automatically handling various protocols including SNA, TCP/IP, and X.25 and supports over 35 platforms. The result is a scalable solution that seamlessly converts from SNA to IP with no product adjustments, as well as seamless integration with existing in-house applications via its exposed API.

## About This Guide

### *About Tivoli Data Exchange*

Tivoli Data Exchange provides the following services:

- Moves and accepts files among all supported platforms
- Provides data compression, if you require it
- Performs binary and ASCII transfers
- Transfers files regardless of their size, format, or destination
- Allows individual status tracking for any phase of the file transfer at any node across the enterprise

Tivoli Data Exchange provides integration capabilities on top of the software, including business process and workflow integration. Exits can be customized by users and can be employed in a plug-and-play manner. They are called at strategic points during a Tivoli Data Exchange transaction.

Two connectors, Multi-File and File-to-Message, allow you to pass data to the Tivoli Data Exchange transfer engine with a simple, consistent application programming interface (API). The data is protected through syncpoint control. Transactional status is distributed across the enterprise. High-performance multiplexing capabilities are employed transparently.

No complex changes, design requirements, or re-engineering efforts are required for existing applications to take advantage of these features.

## **Who Should Read This Guide**

This guide is intended for the following groups:

- Developers using Tivoli Data Exchange to design file-transfer solutions
- Users performing Tivoli Data Exchange solutions
- Tivoli Data Exchange administrators
- MQSeries administrators
- System administrators for the machine on which Tivoli Data Exchange is running

## How This Guide Is Organized

The following table lists and describes the parts and chapters in this manual.

<b>Chapter</b>	<b>Title</b>	<b>Purpose</b>
1	About This Guide	Provides a general introduction to this manual.
2	Tivoli Data Exchange Concepts	Provides an overview of the major components of TDE, its architecture, and how its components work.
3	Tivoli Data Exchange Use and Customization	Provides information on TDE components, starting TDE, security, the options file, and typical data-transfer scenarios.
4	Status Offload Daemon	Describes the status offload component for TDE, how to start it and how to set up the configuration file and command line definitions.
5	Tivoli Data Exchange User Exits	Describes how to use and customize TDE user exits.
6	Security Authorization Exits	Describes how to use security authorization exits and lists technical reference information about the exits and the data structures they use.
7	Working with AS/400 Files	Describes how to transfer AS/400 logical and physical file data.
8	Using the ISPF User Interface	Describes how to access and use ISPF panels to submit and monitor file-transfer requests.
9	Using the 5250 User Interface	Describes how to access and use 5250 panels to submit and monitor file-transfer requests.
10	Using the FTF GUI	Describes how to access and use the TDE Java-based graphical user interface (GUI) to submit and monitor file-transfer requests.
11	Tivoli Data Exchange Explorer	Describes how TDE runs using Microsoft's Management Console (MMC)
12	Directory Monitor (FTFDIRMN)	Describes how to use the TDE Directory Monitor for both UNIX and Windows NT platforms
13	Tivoli Data Exchange Client for 4690	Describes how to install, configure, and start TDE Client for 4690 on each of its supported 4690 platforms.
14	Using Pools	Describes how to use TDE pools to improve file-transfer performance and control the queues used during a file transfer.

Chapter	Title	Purpose
15	Setting Up Configuration Queues	Describes how to work with configuration data in a queue rather than a file.
16	Logging Tivoli Data Exchange Information	Describes how to use both logging options to manage log information written by TDE components during file transfers.

## Contacting Customer Support

We are very interested in hearing from you about your experience with Tivoli products and documentation. We welcome your suggestions for improvements. If you have comments or suggestions about this documentation, please send e-mail to [usib2hpd@vnet.ibm.com](mailto:usib2hpd@vnet.ibm.com).

If you encounter difficulties with Tivoli products, contact Tivoli Customer Support. In the United States, the Tivoli number is 1-800-TIVOLI8 and the IBM number is 1-800-237-5511 (press or say **8** after you reach this number). Both of these numbers direct your call to the Tivoli Customer Support call center. In addition, you can enter <http://www.tivoli.com/support> to view the Tivoli Customer Support home page.

After you link to and submit the customer registration form, you will be able to access many customer support services on the Web. For support services outside the United States and Puerto Rico, contact your local IBM representative or your authorized IBM supplier.

## Where to Look for More Information

Information about MQSeries issues is available in the MQSeries documentation. The IBM corporate website includes a web page that lists the MQSeries manuals and allows you access to an online version of each manual. At publication time, this page's URL was <http://www.software.ibm.com/ts/mqseries/library/manuals>.

## Conventions

The following elements are used in this guide to make it easier to use:

---

### **Note:**

Notes provide additional information about the current subject.

---

---

### **Warning:**

Warnings alert you to situations that can cause problems, such as the loss of data, if you do not carefully follow instructions.

---

---

### **Sidebar**

Sidebars contain information that does not fit specifically with the flow of the current topic, but is important to the topic. Sidebars are usually a short topic.

---

All `syntax`, `operating system terms`, and `literal examples` are presented in this typeface.

*Italics in a command string signify variables.*

Text enclosed in angle brackets (<>) denotes variable information. Replace the variable information with the actual value.



---

---

# Tivoli Data Exchange Concepts

This chapter provides an overview of the Tivoli Data Exchange (TDE) product and its architecture. It includes the following sections:

Section	Page
Introduction	19
Why FTF?	19
Architectural Overview	21

## Introduction

Tivoli Data Exchange (TDE) (FTF) provides the means to exchange information between dissimilar networks, operating systems, databases, and applications. For developers who want to leverage the strengths of MQSeries, FTF is a high-performance solution for distributing mission-critical information throughout the enterprise. Surpassing simple data movement tools, FTF capabilities extend from basic data movement to the comprehensive management of all data-transfer activities. This results in greater productivity and enables users to integrate FTF into existing business processes with minimal impact and/or change.

## Why FTF?

Data-transfer mechanisms available to date have used existing, underlying network transport protocols. These products typically provide a connection-oriented, synchronous data-transfer facility. Because of the inherent nature of the transport protocols and the architecture of the products, every data-transfer request requires a session between the computers involved. Data transfers are usually limited to a point-to-point transfer of a single file or a series

of files between one source and one destination. In some cases, if a transfer fails at any point, the entire transfer session must be reestablished and started again. In addition, no facility for intelligent recovery, checkpoint restarting, load-balancing, or assured data delivery is provided in the event of network or other system failures.

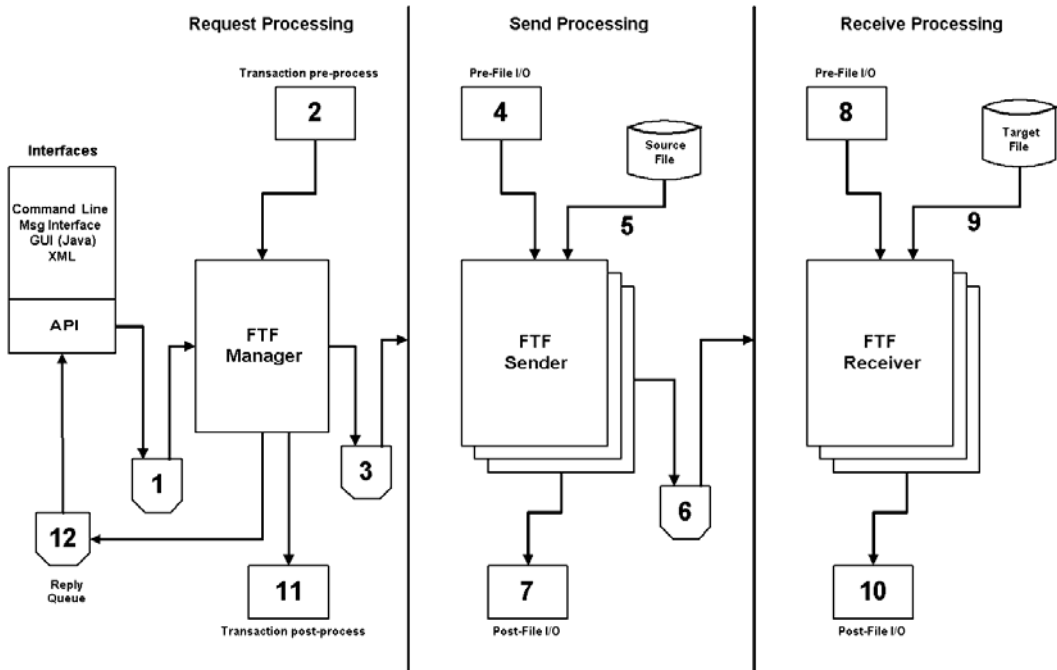
Most traditional data-transfer utilities and third party products do not provide incremental and scalable solutions. For organizations with diverse business application integration requirements and network infrastructures, the traditional data-transfer utilities fall short in delivering a robust solution to solve their unique data-transfer needs today, and providing a migration path to real-time transaction-based highly distributed applications in the future.

Most of the available data-transfer solutions are modeled after the UNIX style File Transfer Protocol (FTP) utility and implement a synchronous data-transfer model between a client and server. Key characteristics of these types of data-transfer implementations are:

- Connection oriented – A network session or conversation is established between the source and the destination. The underlying transport mechanism may be reliable or unreliable. Sessions are created, used, and terminated for each data-transfer request.
- Non-network transparent – By having to interact directly with the network layer protocols, most data-transfer utilities cannot isolate the source and the destination processes from the idiosyncrasies of the network layer.
- Non-modular – In most data-transfer utilities, the source and the destination processes interact directly with the file systems and are directly responsible for manipulating the data on both the source and destination.
- Limited restart and recovery – Most data-transfer utilities provide limited restart and recovery logic to recover from system and network failures. In most cases, the entire data-transfer request must be restarted from scratch in case of system and network failures.
- Limited workflow and application integration – Tools such as FTP provide limited or no work-flow and business-application integration hooks.
- Lack of centralized monitoring and administration – Providing a consistent management and administration framework for the data-transfer utility and the underlying transport mechanism is limited to few data-transfer utilities.

## Architectural Overview

It is important to understand the architecture of FTF in order to grasp the references made throughout this document to the various FTF components. The following diagram illustrates the FTF architecture.



The following major components compose FTF:

- FTF Interfaces
- FTF Manager
- FTF Sender
- FTF Receiver
- FTF Status

The flow of a data transfer occurs as follows:

1. A request is submitted to FTF by one of the supported interfaces. The request is passed to the FTF Manager's input queue.
2. After the FTF Manager accepts the request, but before processing it, the FTF Manager transaction preprocess exit can be called.

3. The FTF Manager submits the request to the FTF Sender via the FTF Sender's input queue.
4. After the FTF Sender accepts the request, but before processing the data being transferred, the FTF Sender can call the sender pre-process exit to perform application-specific processing.
5. The FTF Sender reads and transforms the data into MQSeries messages.
6. The messages that make up the data are submitted to the FTF Receiver via the FTF Receiver's input queue and data queues.
7. After processing the data, the FTF Sender can call the post-process exit to perform application-specific processing.
8. After the FTF Receiver accepts the data, but before processing it, the FTF Receiver can call the receiver pre-process exit to perform application-specific processing.
9. The FTF Receiver retrieves the data messages and processes the data accordingly.
10. After the FTF Receiver processes the data, it can call the FTF Receiver post-process exit to perform application-specific processing.
11. The FTF Manager receives all responses and ends the logical unit of work (LUW). Before ending the LUW, the FTF Manager can call the manager post-processing exit.
12. An optional response is delivered to the appropriate end-user interface indicating that the data transfer has completed.

## **FTF Interfaces**

The FTF interfaces communicate with the FTF subsystem. FTF currently supports the following interfaces:

- command-line interface
- C API
- COBOL API on OS/390
- Java graphical user interface (GUI)
- 5250 user interface on AS/400
- XML scripting

With the exception of XML scripting, the API is the lowest layer for communicating with FTF. The API places the input messages on the FTF input queue and receives replies from a predefined reply queue. The input queue is defined in the FTF configuration file.

## **FTF Manager**

The FTF Manager reads its input queue, creates log entries, submits status messages to the status queues, and manages the state of all transfers. The FTF Manager starts the transfer unit of work and ends the transfer unit of work. The processing FTF Manager is always the originating queue manager (oqm). FTF requests are stored at the FTF Manager for processing. The node on which the FTF Manager executes is considered the originating node. The request is forwarded to the appropriate FTF Sender node as defined by the required input. The message is submitted to the FTF Sender's input queue, which is defined in the FTF configuration file. If the FTF Sender is on a remote queue manager, the message destination is resolved to a transmission queue and transmitted appropriately based on the predefined MQSeries configuration.

The FTF Manager correlates all operational replies and reports the final status of the FTF transactions. These replies should not be confused with the status messages that are part of the integrated status subsystem. Status messages are independent and not required for internal FTF processing. There are four completion codes that the FTF Manager reports:

- Request completed successfully
- Request failed (accompanied with the appropriate failure information)
- Request expired
- Request canceled

---

### **Note:**

For more information about completion codes, see *Tivoli Data Exchange Messages and Codes*.

---

## **FTF Sender**

The FTF Sender reads its input queue, creates log entries, submits status messages, and transforms the data into MQSeries messages. The processing FTF Sender is always where the source data resides and is identified as the source FTF Sender or source queue manager (sqm). The FTF Sender has the capability to read the data and place it in a staging area or transmit it directly to the target FTF Receiver. The request, and all of its associated data, is forwarded to the target FTF Receiver as defined by the required input. If the FTF Receiver is on a remote queue manager, the message destination is resolved to a transmission queue or set of transmission queues and appropriately transmitted via the MQSeries message channel agent. Immediately after completing the processing of its portion of the FTF transaction, the FTF Sender replies to the originating FTF Manager.

## **FTF Receiver**

The FTF Receiver reads its input queue, creates log entries, submits status messages, and receives incoming data from MQSeries to create the target data. The FTF Receiver is always the destination for the data and is identified as the destination FTF Receiver or destination queue manager (dqm). The FTF Receiver accepts a data-transfer request and processes the inbound data from its data queues. Immediately after processing its portion of the FTF transaction, it reconstructs the target data from the MQSeries messages and submits an operational reply to the originating FTF Manager.

## **FTF Status**

The FTF status queues are defined in the FTF configuration file. FTF Status messages, which are submitted by each of the FTF components that make up the FTF subsystem, are not required for internal FTF processing. Rather, they provide a status reporting subsystem that can be exercised by a number of different mechanisms to report on the current and past status of data-transfer requests. Status messages are MQSeries messages destined for the queue or list of queues defined in the FTF configuration file. These queues can be defined as local or remote, and can be processed and viewed with any of the supporting interfaces or user-written programs. These messages may be disabled, as FTF can operate without them (see Note).

---

**Note:**

To disable status messages, you need to change the FTF configuration file, as follows:

- Stop the FTF components.
  - Go to the FTF configuration file (ftfconfig.ini) and find the “Status Defaults” section.
  - Change status to off (Status=OFF). The default status is NOTPERSIST.
  - Restart the FTF components for the change to take effect.
-





---

# Tivoli Data Exchange Use and Customization

This chapter describes how to start and shut down Tivoli Data Exchange (TDE) components, how status information is handled, and how to use the options file. It also describes security and includes typical data-transfer scenarios. The chapter contains the following sections:

Section	Page
FTF Components	28
Starting FTF	32
Getting Status Information from FTF Components	42
Shutting Down FTF Components	46
Security	50
Using the Options File	53
Typical Data-Transfer Scenarios	54

## Assumptions

This chapter makes the following assumptions:

- TDE (FTF) has been successfully installed and configured on each node participating in data-transfer requests.
- The appropriate version of MQSeries is running on each node participating in data-transfer requests.

## **FTF Components**

The FTF system contains seven components: FTF Managers, FTF Senders, and FTF Receivers, FTF Logger, FTF Status, FTF Broker for publish/subscribe, and FTF Directory Monitor. This section describes the FTF Manager, Sender, and Receiver components and contains a summary of these component's processing responsibilities. The other components each have a chapter in this manual to describe their functions. [TRUE FOR LOGGER???

### **The FTF Manager**

The FTF Manager accepts data-transfer requests from the FTF interfaces. The FTF Manager that accepts a request is considered that request's originating queue manager (oqm). The oqm tracks all data-transfer requests throughout the entire transfer process, handling expiration conditions, recovery, and correlation of all replies. A single FTF Manager can service requests from multiple requesting nodes and process requests from multiple source nodes where source files are processed and submitted to a given destination.

## **Request Interfaces**

The request module is the catalyst for a data-transfer request. Access to the module varies from interface to interface, but the processing remains the same: submitting the data-transfer request with all required information.

The following list contains the request module access point for each FTF interface:

- C API – *FTFReq* function in the *Tivoli Data Exchange Technical Reference*, “FTFReq”.
- Command line – *Tivoli Data Exchange Technical Reference*, “FTF”.
- ISPF panels – *Requesting to Transfer Files* panel (See “Using the ISPF User Interface” chapter).
- FTF GUI – *FTF Request Dialog* window (see page 227).
- 5250 panels – *FTF* panel (see page 173).
- COBOL API – *Tivoli Data Exchange Technical Reference*, “COBOL API Reference”.
- XML – *Tivoli Data Exchange Technical Reference*, “XML Integration” and “Tivoli Data Exchange Virtual Machine Connector”.

## **FTF Manager Processing Summary**

When the FTF Manager receives a data-transfer request of any kind on its input queue, it checks the expiration of the request. If the request has expired, the transaction is logged as expired and is completed. If the FTF Manager receives a reply stating that the FTF transaction is complete and a reply is requested, it forwards this information to the interface that made the request. If the request has not expired, the FTF Manager forwards the request to the target FTF Sender, as specified with the source queue manager (sqm).

---

### **Note:**

A request expires after the expiration time has been exhausted. Once the expiration time is exhausted, the FTF Manager flags the request as marked for expiration and waits for a response from the other components. The response from the other components can confirm that the request is expired or that the request completed within the specified expiration time. If a request expires, all messages pertaining to that data-transfer request are purged.

---

The transaction is complete when the FTF Manager receives a transaction-complete notification from the FTF Receiver. The FTF Manager coordinates responses from all the components and if required, responds back to the appropriate interface. In addition, FTF Manager submits status messages to the FTF status subsystem based on the status queues defined in the FTF configuration file.

## **The FTF Sender**

The FTF Sender can process data-transfer requests from any number of FTF Managers. It processes the data being transferred and submits all data as messages to MQSeries for transmission to the destination node.

### **FTF Sender Processing Summary**

The FTF Sender is associated with the source queue manager (sqm) set in the calling interface. Typically, the interface forwards a data-transfer request to a target FTF Manager. The FTF Manager forwards the request to the appropriate FTF Sender. The FTF Sender checks the request before processing to ensure that the expiration time has not been exhausted. If the data-transfer request has not expired, the FTF Sender processes the data and forwards it to the target FTF Receiver. When the data-transfer request is complete, the FTF Sender sends a reply back to the originating FTF Manager.

To support remote staging, the stage queues, FTFSDR.STAGE and FTFSDR.STAGE.CONTROL, are defined as remote queues that resolve to a local queue on another queue manager. The source file then stages at the remote queue manager. To send the files in the remote stage queues, define the sqm on the request as the queue manager where the file is actually staged. The FTF Sender, however, cannot purge or query files. These functions must be performed by the FTF Receiver. If purge or query is attempted with an FTF request on the remote stage queue, the Sender reports a failure attempting to open the remote queue.

Throughout its piece of the process, the FTF Sender submits status messages to the status queues defined in the FTF configuration file.

## **The FTF Receiver**

The FTF Receiver can process inbound data-transfer requests from multiple FTF Senders. The FTFReceiver accepts the request, pulls the data from MQSeries, and reconstructs it according to the rules dictated by the input of the data-transfer request.

## **FTF Receiver Processing Summary**

The FTF Receiver is associated with the destination queue manager (dqm) set in the calling interface. Typically, the interface forwards a data-transfer request to a target FTF Manager. The FTF Manager forwards the request to the target FTF Sender. The FTF Sender processes the data and forwards it to the target FTF Receiver. The FTF Receiver checks the request before processing to ensure that the expiration time is not exhausted. If the data-transfer request has not expired, it is processed by the FTF Receiver. When the data transfer is complete, the FTF Receiver sends a reply back to the originating FTF Manager.

In addition, the proper status messages must be submitted to the FTF status subsystem based on the status queues defined in the FTF configuration file. The FTF Receiver receives incoming requests and processes the file according to the attributes specified at input.

---

### **Note:**

When a data-transfer request fails and the file mode for the FTF Receiver is set to “Create” or “Replace”, the FTF Receiver component deletes the data. If the data-transfer request fails before the FTF Receiver starts processing to the target file, clean-up is not required. However, if the mode is set to “Create” or “Replace”, and the FTF Receiver has started writing data to the target file and the data-transfer request fails, the FTF Receiver deletes the file.

---

## **Starting FTF**

Before you can use FTF to perform a data-transfer request, you must start the components that will be part of that request. One FTF Manager must be running for a file transfer to take place. You must start the following components (at a minimum) before you can execute a data-transfer request:

- One FTF Manager
- At least one FTF Sender
- At least one FTF Receiver

The methods for starting the FTF components vary depending on the environment in which you are starting the components. This section describes how to start FTF components on the following environments:

- OS/390 (see page 33)
- Win 32 (see page 34)
- AS/400 (see page 36)

- UNIX (see page 38)

## Starting FTF on OS/390 Systems

On OS/390 systems, you can use the FTFSTART command to start all components at once.

The FTFSTART command has the following syntax:

```
FTFSTART -sysoutclass classchar -lqm localQueueMgr
        [-cq configQueueName | -cfile configFile]
        -ofile optionsFile
```

Where:

- **-sysoutclass *classChar*** – Contains the one-character sysout class designation.
- **-lqm *localQueueMgr*** – Determines the queue manager from which the FTFSTART command is issued. This value is required on OS/390 systems. On all other systems, if it is not specified, the command attempts to connect to the default queue manager specified.
- **-cq *configQueueName*** – Determines the queue from which the configuration information is to be retrieved for this FTF instance on this node. If no cfile value is specified, this value must be specified. The cq argument points FTF to the queue name rather than to the standard configuration file.
- **-cfile *configFile*** – Contains the fully qualified path and filename for the FTF configuration file. If no cq argument is specified, this value must be specified.
- **-ofile *optionsFile*** – Contains the fully qualified path and filename of a text file used to contain command-line arguments for the FTFSTART command. In the options file, you can set any of the command-line arguments that can be set for the FTFSTART command. Any values specified on the command line override the values in the options file.

For complete information about the FTFSTART command, see “FTFSTART” in the *Tivoli Data Exchange Technical Reference*.

### Setting Security Authorization Values

You can also use the FTFSTART command to set security authorization values. You can set these values in addition to any other values you set on the FTFSTART command line.

In the following example, FTF is started from the queue manager named PROD11A, the configuration file is set to FTFV221.FTF.INI, and the sysoutclass is set to H. The security authorization options are set in the options file named FTFV221.FTF.OPTS.

```
FTFSTART -lqm PROD11A -cfile FTFV221.FTF.INI -ofile  
FTFV221.FTF.OPTS -sysoutclass H
```

### Starting Components Individually

You can also use individual commands to start FTF components. For more information, see “Starting Components Individually” on page 38.

## Starting FTF on Win 32

When you install TDE on a Win 32 machine, an FTF service component is installed. When the service component is started, it starts all FTF components.

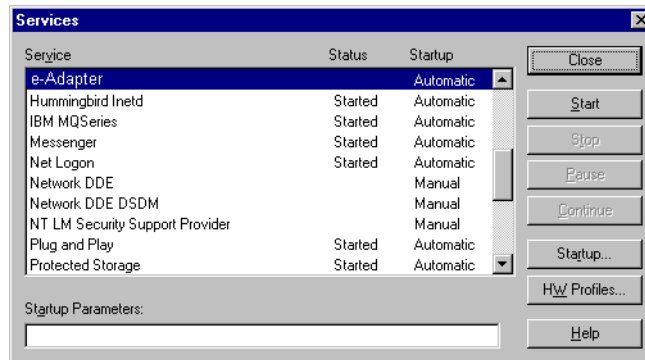
The FTF service can start more than one instance of FTF. An instance is a complete, self-contained set of FTF components. Instances allow you to have a set of FTF components for each environment you support. For example, if you have a test environment and a production environment, the FTF service can be configured to start instances for each environment (for more information about configuring the FTF service, see “Configuring Clients” in the *Tivoli Data Exchange Installation Guide*).

Like all Win 32 services, FTF can be set to start automatically when the machine on which it is running starts. It can also be set to start manually.

To manually start the FTF service component, follow these steps:

1. Open the Windows control panel.
2. Double-click the *Services* icon. The *Services* window appears.





If *Started* appears in the **Status** column, the FTF service component is already started.

3. If no status appears, highlight the FTF line by clicking on it and click the **Start** button. The FTF service component is started.

## Starting Components Individually

You can also use individual commands to start FTF components. For more information, see “Starting Components Individually” on page 38.

### Starting Individual Components in Win

#### 32

If you start an FTF component from the command line, it does not show up in the Services window as started. When you stop the FTF service, however, all components shut down, including those started from the command line.

When you restart the FTF service, the components you started from the command line are not restarted with the service. If you want to use them, you must restart them from the command line or modify the FTF configuration file and FTF initialization file, as necessary, to start them as part of the service. For more information about the FTF service component, see “Installing FTF on Win 32” in the *Tivoli Data Exchange Installation Guide*.

## Starting FTF on AS/400

On the AS/400 platform, the FTF Manager, Sender, Receiver, and Logger are run as batch jobs and are automatically submitted to JOBQ (*user-defined*). The job description is JOBD (*user-defined*).

### Starting an AS/400 Session

To start a session in AS/400, enter the following command at the command prompt:

```
STRCMTCTL LCKLVL (*ALL) CMTSCOPE (*JOB)
```

This command lasts for the entire FTF session and allows you to successfully run any FTF command.

### Using the AS/400 FTF Menu

On AS/400, the FTF menu is used to run FTF components. To access the FTF menu, enter the following command:

```
GO CMDTDEMQ
```

The FTF menu appears and lists options for running the FTF program. The column to the left of the screen lists the processing commands and the component commands by number. The column to the right of the screen lists the corresponding commands. To choose a command, perform one of the following steps:

- Type the number of the command and press **[ENTER]**.
- *Or* -
- Type the command and press **[F4]**. The corresponding command screen appears.

### Starting the FTF Manager

To start the FTF Manager, perform one of the following steps:

- Type **8** and press **[ENTER]**.
- *Or* -

- Type **STRFTFMGR** at the command line and press **[F4]**.

A panel appears listing the input fields for FTF Manager arguments. Enter the appropriate values.

Because information written to the job log is buffered, you should specify a log file. All log messages are immediately written to the log file where they can be viewed. The panel submits a job called FTFMGR.

## **Starting the FTF Sender**

To start the FTF Sender, perform one of the following steps:

- Type **9** and press **[ENTER]**.

**- Or -**

- Type **STRFTFSDR** at the command line and press **[F4]**.

A panel appears listing input fields for FTF Sender arguments. Enter the appropriate values.

Because information written to the job log is buffered, you should specify a log file. All log messages are immediately written to the log file where they can be viewed. The panel submits a job called FTFSDR.

## **Starting the FTF Receiver**

To start the FTF Receiver, perform one of the following steps:

- Type **10** and press **[ENTER]**.

**- Or -**

- Type **STRFTFRCV** at the command line and press **[F4]**.

A panel appears listing input fields for FTF Receiver arguments. Enter the appropriate values.

Because information written to job log is buffered, you should specify a log file. All log messages are immediately written to the log file where they can be viewed. The panel submits a job called FTFRCV.

To call FTF components, type the command and press **[F4]** to be prompted.

## Starting the FTF Logger

To start the FTF Logger, perform one of the following steps:

- Type **11** and press **[ENTER]**.

**- Or -**

- Type **STRFTFLOG** at the command line and press **[F4]**.

A panel appears listing input fields for FTF Logger arguments. Enter the appropriate values. The panel submits a job called FTFLOG.

## Starting FTF in UNIX

You must start FTF components individually on UNIX platforms. For information about starting FTF components individually, see “Starting Components Individually” on page 38.

## Starting Components Individually

In OS/390, Win 32, and UNIX platforms, you can start each type of FTF component individually.

This section lists the commands used to start the following TDE components:

- FTF Managers (see page 28)
- FTF Senders (see page 31)
- FTF Receivers (see page 31)

## Starting the FTF Manager

The FTFMGR command starts the FTF Manager, according to the conditions set in command-line arguments that you provide. The FTFMGR command has the following syntax:

```
FTFMGR -lqm localQueueManager  
        [-cq configQueueName | -cfile configFile]  
        -ofile optionsFile -lfile logFile
```

Where:

- **-lqm *localQueueMgr*** – Determines the queue manager that the FTF Manager connects to. This value is required on OS/390 systems. On all other systems, if it is not specified, the command attempts to connect to the default queue manager specified.
- **-cfile *configFile*** – Contains the fully qualified path and filename for the FTF configuration file. On OS/390 platforms, if no cq argument is specified, this value must be specified. In other operating systems, it is optional if the FTF\_CONFIG\_FILE environment variable is set.
- **-cq *configQueueName*** – Determines the queue from which the configuration information is to be retrieved for this FTF instance on this node. On OS/390 platforms, if no cfile value is specified, this value must be specified. The cq argument points FTF to the queue name rather than to the standard configuration file.
- **-ofile *optionsFile*** – Contains the fully qualified path and filename of a text file used to contain command-line arguments for the FTFMGR command. In the options file, you can set any of the command-line arguments that can be set for the FTFMGR command. Any values specified on the command line override the values in the options file.
- **-lfile *logFile*** – Contains the fully qualified path and filename of a log file to which the FTF Manager writes. Although log files are not supported on OS/390, the information is written to the standard SYSOUT queue.

For complete information about the FTFMGR command, see “FTFMGR” in the *Tivoli Data Exchange Technical Reference*.

In the following example, the FTF Manager connects to queue manager PROD11A, according to settings in the FTFV230.FTF.INI file. Log entries are written to the FTFV230.MGR.LOG file.

```
FTFMGR -lqm PROD11A -cfile FTFV230.FTF.INI -lfile  
FTFV230.MGR.LOG
```

## Starting the FTF Sender

The FTFSDR command starts the FTF Sender, according to the conditions set in command-line arguments that you provide. The FTFSDR command has the following syntax:

## Tivoli Data Exchange Use and Customization

### Starting FTF

```
FTFSDR -lqm localQueueManager  
[-cq configQueueName | -cfile configFile]  
-ofile optionsFile -lfile logFile
```

Where:

- **-lqm localQueueMgr** – Determines the queue manager that the FTF Sender connects to. This value is required on OS/390 systems. On all other systems, if it is not specified, the command attempts to connect to the default queue manager specified.
- **-cfile configFile** – Contains the fully qualified path and filename for the FTF configuration file. On OS/390 platforms, if no cq argument is specified, this value must be specified. In other operating systems, it is optional if the FTF\_CONFIG\_FILE environment variable is set.
- **-cq configQueueName** – Determines the queue from which the configuration information is to be retrieved for the FTF instance on this node. On OS/390 platforms, if no cfile value is specified, this value must be specified. The cq argument points FTF to the queue name rather than to the standard configuration file.
- **-ofile optionsFile** – Contains the fully qualified path and filename of a text file used to contain command-line arguments for the FTFS DR command. In the options file, you can set any of the command-line arguments that can be set for the FTFS DR command. Any values specified on the command line override the values in the options file.
- **-lfile logFile** – Contains the fully qualified path and filename of a log file to which the FTF Sender writes. Although log files are not supported on OS/390, the information is written to the standard SYSOUT queue.

For complete information about the FTFS DR command, see “FTFS DR” in the *Tivoli Data Exchange Technical Reference*.

In the following example, the FTF Sender connects to queue manager PROD11A, according to settings in the FTFV230.FTF.INI file. Log entries are written to the FTFV230.SDR.LOG file.

```
FTFSDR -lqm PROD11A -cfile FTFV230.FTF.INI -lfile  
FTFV230.SDR.LOG
```

## Starting the FTF Receiver

The FTFRVCV command starts the FTF Receiver, according to the conditions set in command-line arguments that you provide. The FTFRVCV command has the following syntax:

```
FTFRVCV -lqm localQueueManager
        [-cq configQueueName | -cfile configFile]
        -ofile optionsFile -lfile logFile
```

Where:

- **-lqm *localQueueMgr*** – Determines the queue manager that the FTF Receiver connects to. This value is required on OS/390 systems. On all other systems, if it is not specified, the command attempts to connect to the default queue manager specified.
- **-cfile *configFile*** – Contains the fully qualified path and filename for the FTF configuration file. On OS/390 platforms, if no cq argument is specified, this value must be specified. In other operating systems, it is optional if the FTF\_CONFIG\_FILE environment variable is set.
- **-cq *configQueueName*** – Determines the queue from which the configuration information is to be retrieved for this FTF instance on this node. On OS/390 platforms, if no cfile value is specified, this value must be specified. The cq argument points FTF to the queue name rather than to the standard configuration file.
- **-ofile *optionsFile*** – Contains the fully qualified path and filename of a text file used to contain command-line arguments for the FTFRVCV command. In the options file, you can set any of the command-line arguments that can be set for the FTFRVCV command. Any values specified on the command line override the values in the options file.
- **-lfile *logFile*** – Contains the fully qualified path and filename of a log file to which the FTF Receiver writes. Although log files are not supported on OS/390, the information is written to the standard SYSOUT queue.

For complete information about the FTFRVCV command, see “FTFRVCV” in the *Tivoli Data Exchange Technical Reference*.

## Tivoli Data Exchange Use and Customization

### *Getting Status Information from FTF Components*

In the following example, the FTF Receiver connects to queue manager PROD11A, according to settings in the FTFV230.FTF.INI file. Log entries are written to the FTFV230.RCV.LOG file.

```
FTFRCV -lqm PROD11A -cfile FTFV230.FTF.INI -lfile  
FTFV230.RCV.LOG
```

## Getting Status Information from FTF Components

The FTF status subsystem manages messages that describe the status of a FTF data-transfer request. These status messages can be generated by FTF or from within any user exit or security authorization module. This section lists and describes the status messages generated by each type of FTF component.

To access status information use one of the following methods:

- The FTFSTAT panel in the ISPF interface (see “Viewing Status Information” on page 155).
- The status display in the FTF GUI (see “Viewing Data-Transfer Status Information” on page 229).
- The FTFSTAT command (see the *Tivoli Data Exchange Technical Reference*, “FTFSTAT”).
- The *FTFStatusGetDetailList* (see the *Tivoli Data Exchange Technical Reference*, “FTFStatusGetDetailList”) and *FTFStatusGetSummaryList* (see the *Tivoli Data Exchange Technical Reference*, “FTFStatusGetSummaryList”) functions in the FTF C API.
- The COBOL API functions of FTFDL (see the *Tivoli Data Exchange Technical Reference*, “FTFDL”) and FTFSFDL (see the *Tivoli Data Exchange Technical Reference*, “FTFSDL”).
- 5250 interface function of FTF Status.

For information about submitting status information from security authorization modules, see the *Tivoli Data Exchange Messages and Codes* manual.

For information about submitting status information from user exit modules, see the “User Exits” chapter in this manual.



## The FTF Manager and the Status Subsystem

Each status message is submitted to the user-defined status queue(s). The FTF status subsystem manages these messages.

Each message contains a timestamp that is generated when the activity associated with the status message takes place. For example, the time stamp in the REQUEST\_EXPIRED message is the timestamp at the moment the FTF Manager discovered the condition.

The following messages are generated by the FTF Manager and submitted to the FTF status subsystem:

- **Request Expired** – Indicates that a data-transfer request has expired. This condition can occur when the FTF Manager begins processing a new request but the request's expiration time has been exceeded. The request is terminated immediately and a reply is generated, if specified. This message is also used when the FTF Manager flags a request for expiration and receives a confirmation response back from the FTF Receiver or FTF Sender that the request has expired.
- **Request Failed** – Indicates that a data-transfer request cannot be successfully completed and is logged as a failed transaction. This message is issued when any FTF component detects that a data-transfer request did not complete successfully. An associated internal processing code with an optional sense code from any external components is issued with this message. For more information on codes, see *Tivoli Data Exchange Messages and Codes*.
- **Request Submitted** – Indicates that the FTF Manager has successfully submitted the request to the target FTF Sender. This message signifies that the request was successfully submitted to MQSeries for delivery to the target FTF Sender.
- **Request Completed** – Indicates that the FTF Manager has received a response from the FTF Receiver that the data being transferred has been processed. The FTF Manager submits a message to the status subsystem that the request is complete.
- **Request Canceled** – Indicates that the FTF Manager has validated that a cancellation notice has been received and all processing on the request has stopped.

## **Tivoli Data Exchange Use and Customization**

### *Getting Status Information from FTF Components*

- **Request Flagged for Cancel** – Indicates that the FTF Manager has received a request to cancel an FTF data-transfer request and has submitted cancel directives to other components. Until confirmations are returned, the request is flagged for cancellation.
- **Staging Completed** – Indicates that the FTF Manager has received a message from the FTF Sender that the request to send the data to the staging queue only is complete.

## **FTF Sender and the Status Subsystem**

Each status message listed in this section is submitted to the status queue(s) by the FTF Sender. The FTF status subsystem manages these messages. Each message contains a timestamp that is generated when the activity in the status message takes place.

The following messages are generated by the FTF Sender and submitted to the FTF status subsystem:

- **Request Received** – Indicates that the FTF Sender has accepted a new data-transfer request. When the FTF Sender receives a data-transfer new request, this message is submitted to the status subsystem.
- **Request Expired** – Indicates that a request has expired. This condition occurs when the FTF Sender begins or tries to continue processing a data-transfer request for which the expiration time has been exceeded.
- **Request Failed** – Indicates that a data-transfer request cannot be successfully completed. The message contains an associated internal processing code with an optional sense code from any external components. In addition, optional text may be included.
- **Request Transmitted** – Indicates that the FTF Sender has completed processing a specified data-transfer request. This message signifies that the entire request and the data has been submitted to MQSeries for transmission to the target FTF Receiver.
- **Request Cancelled** – Indicates that the FTF Sender has received a cancellation request and stopped processing the specified FTF data-transfer request.
- **Processing** – Indicates the number of messages which are generated to compose the file can be reported as status. This message is generated while the FTF Sender processes a file.

- **Request Recovering** – Indicates that the FTF Sender has detected that a previous data-transfer request has failed and recovery is required and has begun.

## **FTF Receiver and the Status Subsystem**

Each status message listed in this section is submitted to the status queue(s) by the FTF Receiver. The FTF status subsystem manages these messages. Each message contains a timestamp that is generated when the specific activity in the status message has taken place.

The following messages are generated by the FTF Sender and submitted to the FTF status subsystem:

- **Request Received** – Indicates that the FTF Receiver has accepted a new request. When the FTF Receiver receives a data-transfer request, it submits this message to the status subsystem.
- **Request Expired** – Indicates that a request has expired. This condition occurs when the FTF Receiver begins or tries to continue processing a data-transfer request for which the expiration time has been exceeded.
- **Request Failed** – Indicates that a transfer cannot be successfully completed. An associated internal processing code with an optional sense code from any external components is included. In addition, optional text can be included.
- **Request Completed** – Indicates that the request and its associated file have been processed and a response has been submitted to MQSeries for transmission to the originating FTF Manager. This message is generated when the FTF Receiver has completed processing the data-transfer request.
- **Request Cancelled** – Indicates that the FTF Receiver has received a cancellation request and stopped processing the specified FTF data-transfer request.
- **Processing** – Indicates the number of messages that are generated to compose the file can be reported as status. This message is generated while the FTF Receiver processes a file.
- **Request Recovering** – Indicates that the FTF Receiver has detected that a previous transaction has failed and recovery is required and has begun.

## Shutting Down FTF Components

This section describes how to shut down FTF components using the following methods:

- The FTFEND command
- The FTFSTART address space commands (see “Using FTFSTART Address Space Commands” on page 48)

The following sections also list instructions for shutting down FTF components:

- The FTFEND panel in the ISPF interface (see “Shutting Down Components” on page 164)
- The FTF End panel in the 5250 interface (see “Using the FTFEND Command” on page 46)
- The FTF End option in the FTF GUI (see “Shutting Down Tivoli Data Exchange Components” on page 257)
- The FTFShutdown C API (see “FTF Shutdown” in the *Tivoli Data Exchange Technical Reference*)
- The FTFEND COBOL API (see “FTFEND” in the *Tivoli Data Exchange Technical Reference*)

## Using the FTFEND Command

The FTFEND command allows you to shut down specific FTF components on all platforms that support command-line processing. It typically has the following syntax:

```
FTFEND -lqm localQueueMgr [-cpt compNum | -component compNum]  
      -all -timeout expireVal [-immediate | -quiesce]  
      [-cfile configFile | -cq configQueueName] -ofile optionsFile
```

Where:

- **-lqm *localQueueMgr*** – Determines the queue manager that the FTFEND command connects to. This value is required on OS/390 systems. On all other systems, if it is not specified, the command attempts to connect to the default queue manager specified.

- **[-cpt compNum | -component compNum]** – Specifies the component being shut down. Use either the -cpt argument or the -component argument.

**Valid values:**

- 1** – FTF Manager
  - 2** – FTF Sender
  - 3** – FTF Receiver
  - 4** – FTF logging
  - 5** – FTF status
  - 6** – FTF broker
  - 7** – FTF directory monitor
- **-all** – Shuts down all FTF components that are running.

---

**Note:**

If neither **-cpt** nor **-all** argument is entered, then FTFEND shuts down the FTF Manager, Sender, and Receiver by default.

---

- **-timeout expireVal** – Determines the amount of time until the FTFEND times out. If the time limit is exceeded, the specified component still shuts down, but also generates an error message. This value is measured in seconds. **Valid values:** 1-32767
- **-immediate** – Shuts the component down immediately, before completing any current work. If you specify this argument, you cannot specify the quiesce argument.
- **-quiesce** – Shuts the component down after its work is completed. If you specify this argument, you cannot specify the immediate argument.
- **-cfile configFile** – Contains the fully qualified path and filename for the FTF configuration file. On OS/390 platforms, if no cq argument is specified, this value must be specified. In other operating systems, it is optional if the FTF\_CONFIG\_FILE environment variable is set.
- **-cq configQueueName** – Determines the queue from which the configuration information is to be retrieved for this FTF instance on this node. On OS/390 platforms, if no cfile value is specified, this value must be specified. The cq argument points FTF to the queue name rather than to the standard configuration file.

## Tivoli Data Exchange Use and Customization

### *Shutting Down FTF Components*

- **-ofile optionsFile** – Contains the fully qualified path and filename of a text file used to contain command-line arguments for the FTFEND command. In the options file, you can set any of the command-line arguments that can be set for the FTFEND command. Any values specified on the command line override the values in the options file.

For complete information about FTFEND syntax, see “FTFEND” in the *Tivoli Data Exchange Technical Reference*.

## Using FTFSTART Address Space Commands

The FTFSTART command includes multiple facilities for shutting down FTF components on OS/390. They include:

- The FTFEND facility that stops the FTF components running under the FTFSTART address space. When all the components have terminated, the FTFSTART address space terminates automatically.
- The CONSOLE interface commands provided by the FTFSTART facility.

This section describes the CONSOLE interface commands provided by FTFSTART.

### The PURGE Command

The FTFSTART address space treats the PURGE command as if it were an FTFEND request. The FTFSTART command issues an FTFEND request to all the components that are currently running. FTFSTART waits two minutes before shutting down the components, so all components can respond with a shutdown reply. If they all reply within two minutes, FTFSTART shuts them down and terminates itself. If any of the components do not reply within two minutes, the PURGE command waits until they reply before it shuts down.

The PURGE command has the following syntax:

**P** *addressSpaceName*

Where *addressSpaceName* is the name of the FTFSTART address space.

## The MODIFY Command

The MODIFY command issues a variety of requests that require passing information in command strings to the FTFSTART module. It allows you to perform the following tasks:

- Display a list of the tasks running in the FTFSTART address space.
- Shut down FTF components under the same provisions as the Purge command.
- Shut down FTF components immediately, without regard to their current work.

The MODIFY command has the following syntax:

*F addressSpaceName, commandQualifier*

Where:

- *addressSpaceName* is the name of the FTFSTART address space.
- *commandQualifier* dictates the operation you want the MODIFY command to perform. The following table lists and describes its valid values.

Command Qualifier	Description
DISPLAY [ACTIVE]	Displays all the tasks currently running in the current address space.
STOP	Stops all FTF components according to the conditions specified for the PURGE command.
STOP MODE(FORCE)	Immediately terminates all FTF components running in the address space. You should use this option only under emergency conditions, such as a runaway process. If you cannot stop FTF with the PURGE command or the STOP qualifier, use this command qualifier.

---

### Note:

If you use the MODE(FORCE) command qualifier, you may not be able to recover current data-transfer requests. Typically, you should use the FTFEND function provided in each interface, the PURGE command, or the MODIFY STOP command to shut down FTF components.

---

## Security

FTF currently supports MQSeries object-level security. If appropriate access is enabled for the queue objects to which FTF requires access, FTF abides by this security, provided that OAMSecurity is enabled as specified in the FTF configuration file.

The FTFconfiguration file includes the following stanza for OAMSecurity:

OAMSecurity=*securitySetting*

Where *securitySetting* determines the whether OAMSecurity is enabled. **Valid values:** ON, OFF

During the execution of a data-transfer request, various FTF components check this value.

If OAMSecurity is enabled before a request is submitted FTF retrieves the user name of the requester. The user name is passed along with other user identity information and used by all MQSeries operations to read from and write to FTF queues. Each of the FTFcomponents in the subsystem (FTF Managers, FTF Senders, and FTF Receivers), upon receiving a request, determine whether OAM Security is enabled from its active FTF configuration. If OAM Security is enabled, FTF ensures that a valid user name has been provided. If the user name is invalid, the request is rejected.

FTF provides the ability to direct which queues data may span. Therefore, various security schemes can be generated based on a data-transfer request. For example, a user may be granted certain rights to a given FTF pool. In addition, various exits can be implemented to provide additional authority verification and authentication.

## OAM Security on OS/390

To implement OAM Security on OS/390, the RACF security administrator, MQSeries systems programmer, and FTF administrator need to be involved. Perform the following steps to ensure that OAM Security will work correctly.

1. Set OAMSecurity=ON in the FTF configuration file.
2. Ensure that RACF MQADMIN security is on.
3. In the MQADMIN class, ensure that ALTERNATE user security is on.

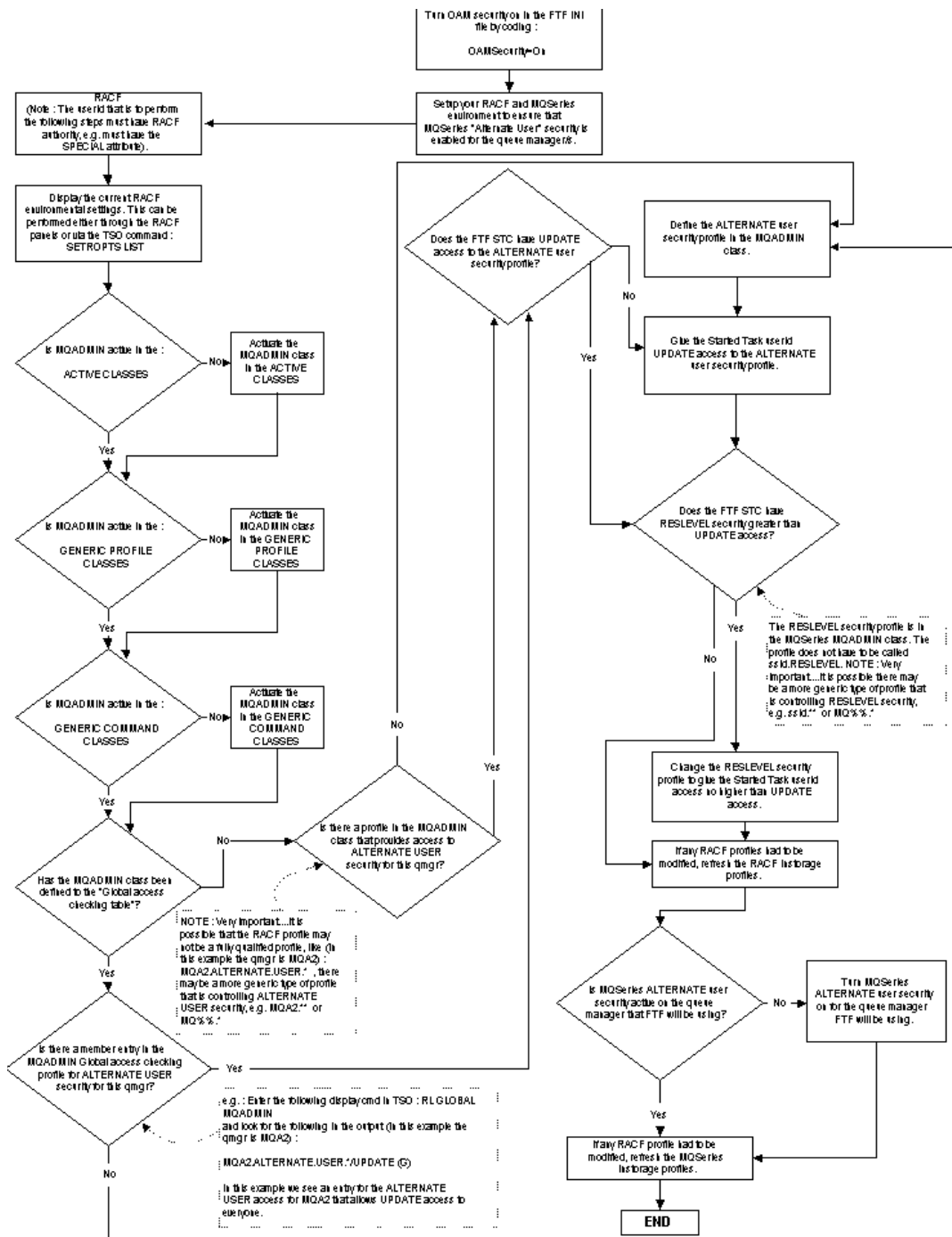


4. GRANT the TDE Started Task userid UPDATE access to ALTERNATE user security.
5. Ensure that the TDE Started Task userid does **not** have RESLEVEL security higher than UPDATE. (If it does, alternate user security will be bypassed.)
6. Refresh RACF security profiles.
7. Ensure that MQSeries Alternate security is turned on.
8. Refresh MQSeries security.
9. Stop and restart the TDE Started Task.

The following flowchart illustrates these steps in more detail.

# Tivoli Data Exchange Use and Customization

## Security



## Using the Options File

The options file is a text file used to store FTF arguments processed by the command line. The options file is referenced with the `-ofile fileName` argument. The options file contains one argument per line and can contain comments that start with an asterisk (\*) in the first column of a new line.

Each FTF command allows you to use the `ofile` argument to specify an options file. In the following example, the FTF command uses an options file:

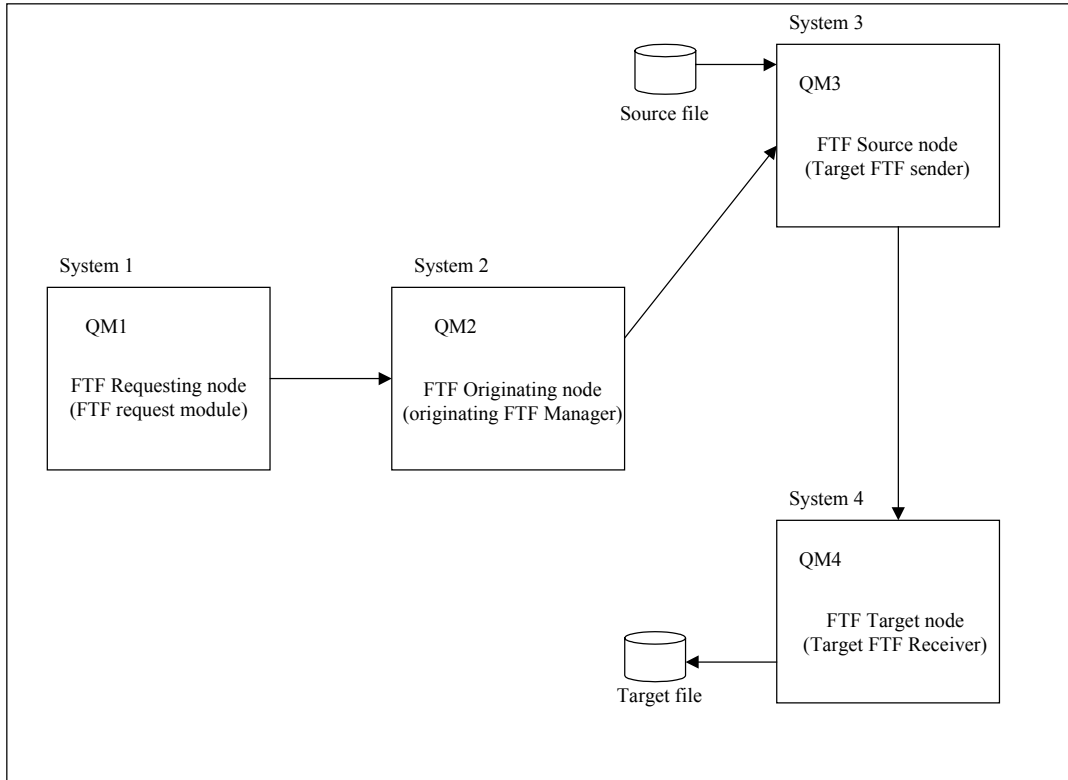
```
FTF -cfile c:\ftf\ftfconfig.ini  
    -ofile c:\ftf\ftfopt.txt
```

The following example is an options file used for a data-transfer request executed using the FTF command.

```
-SQM SHADOW1  
-SPATH C:\DATA\FILE1.FTF  
-DQM SHADOW1  
-DPATH C:\OUT\FILE.3  
  
*pre process  
-exit 3  
-exitdll exit1  
-exitentry exit1  
-exitdata "c:\temp\pre.bat AUDIT_START"  
  
-exit 4  
-exitdll exit1  
-exitentry exit1  
-exitdata "c:\temp\post.cmd"  
-mode replace  
-compress
```

# Typical Data-Transfer Scenarios

The following diagram is used for the FTF command-line examples to demonstrate how to perform data transfers.



## Scenario One

The goal of this example is to send a file between System 3 and System 4 and the following assumptions/statements are made:

- MQSeries has been configured according to FTF requirements.
- The FTF Manager, FTF Sender, and FTF Receiver have all been started.
- The data-transfer request is entered on System 1 using queue manager QM1.
- The FTF Manager that controls the request is on System 2 using queue manager QM2.
- The FTF Sender accesses the source file on System 3 and uses queue manager QM3, which is running on Win 32.
- The FTF Receiver processes the target file on System 4 and uses queue manager QM4, which is running on UNIX.

---

### **Note:**

Any of the components (the FTF Manager, the FTF Sender, or the FTF Receiver) can operate on any of the four systems depicted with the appropriate command-line arguments modified to specify their respective location.

---

The following request could be entered to perform a file transfer.

```
FTF -lqm QM1 -oqm QM2 -sqm QM3 -dqm QM4 -spath  
c:\temp\test.dat -dpath /tmp/test.file.dat
```

---

### **Note:**

If you wish to use the default queue manager on platforms where MQSeries supports the default queue manager, the lqm value is optional.

---

## **Scenario Two**

Scenario Two is based on Scenario One with the following additional assumptions and statements.

- The request takes place on System 3 using queue manager QM3.
- The originating manager is operating on System 3 using queue manager QM3.
- The destination is System 4 using queue manager QM4.
- QM3 is not on OS/390 .
- QM4 is on OS/390 (to illustrate various OS/390 attributes on the command line).

The following request can be entered to perform a data transfer.

```
FTF -sqm QM3 -dqm QM4 -spath c:\temp\test.dat -dpath  
DATASET.TEST.DAT -org PS
```

---

---

# Status Offload Daemon

This chapter describes the status offload daemon component for Tivoli Data Exchange (TDE), which pulls status messages from MQSeries, formats them into XML data streams, and passes them to a user-defined exit for output.

It contains the following sections:

Section	Page
Overview	58
Starting and Stopping the Status Offload Component	60
Configuration File and Command-Line Definitions	61
Recommended Database Schema	65

## Assumptions

This chapter makes the following assumptions:

- Tivoli Data Exchange (TDE) (FTF) has been successfully installed and configured on each node participating in data-transfer requests.
- The appropriate version of MQSeries is running on each node participating in data-transfer requests.

## **Overview**

To efficiently manage a large number of status messages on Server platforms, a new program component has been added to FTF. This new component, a status daemon (FTFSTATD), pulls individual control and detail status messages from MQSeries, formats them into XML data streams, and passes them to an exit for output. The exit must be supplied by the user. On Win 32, a sample exit takes the status messages from the FTF status daemon and exports them to an ODBC relational database (also user supplied). The enableNet Business Process Integrator (BPI) is used to parse the XML data streams for insertion into the supplied database. BPI libraries are included automatically when you install FTF.

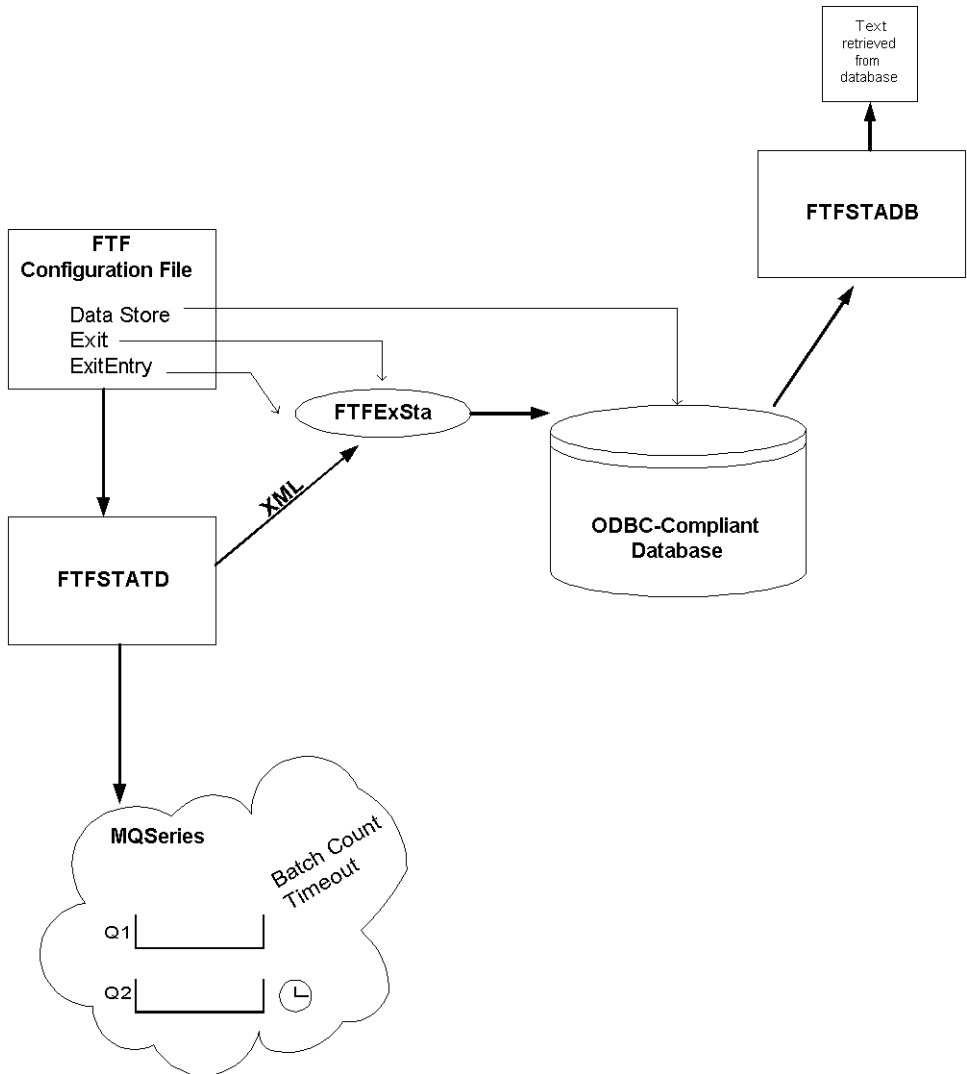
In keeping with FTF conventions, this exit provides a solution that is both high performance and scalable. Using XML data streams supplies a template to handle FTF control and detail status messages, which can then be converted to your chosen format. Your exit consumes the XML data streams based on a set of predefined XML tags that relate to the information contained in the status messages. Even when a new version of FTF is released, you need not change your exit process because both FTF and your exit are using the same XML tags.

The sample FTFExSta, which contains one sample per platform, is included on the CD-ROM. The sample for all FTF Server platforms allows you to write status messages to a flat file. Its entry point is FTFStatusPrint; its data source is the filename. This allows for output of XML data streams to a user-supplied file. The second entry point, for Win 32, outputs the status messages to an ODBC-compliant database. Its entry point is FTFExitStat; its data store is a user-supplied database.

A new command line, FTFSTADB, demonstrates the ability to retrieve and correlate the status information. It accepts the same parameters as the FTF status command line FTFSTAT, including filtering, format, and other options. You can then retrieve status information from the relational database that was offloaded by FTFSTATD.



The following diagram shows the basic architecture and data flow of the FTF's status offload daemon:



## Status Offload Daemon

### *Starting and Stopping the Status Offload Component*

## Starting and Stopping the Status Offload Component

The status daemon component needs to be started only once. It remains operational until you shut it down. If you have defined a default queue manager, the status daemon starts with the FTFSTATD command. If you have not defined a default queue manager, then you need to specify a local queue manager for the status daemon, as follows:

```
FTFSTATD -lqm localQueueMgr
```

Where:

**-lqm localQueueMgr** – The local queue manager that manages the status message queues upon which FTFSTATD operates. If the local queue manager is not specified, the command connects to the default queue manager that is set in the MQSeries configuration.

## Starting the FTFStatD Component from the Win 32 Service

When using the Win 32 Service, you can set the status offload component to start automatically with the other components by placing the following command line in the FTF.INI configuration file:

```
statd.commandline = -lfile E:\logs\ftfstatd.log
```

Where:

**-lfile logFile** – Name of the log file where the status daemon writes error messages.

If the StatusOffloadNumInstances value is greater than 1, then the command line should use the same numbering convention as the FTF Sender and Receiver command lines: i.e., statd.commandline1, statdcommandline2, etc.

## Stopping the FTFSTATD Component from the Win 32 Service

When using the Win 32 Service, you can stop the status offload component without stopping FTF first. To do this, issue the following command:

```
ftfend -lqm localQueueMgr -cpt 5
```

Where:

**-lqm** *localQueueMgr* – The local queue manager that manages the status message queues upon which FTFSTATD operates. If the local queue manager is not specified, the command connects to the default queue manager that is set in the MQSeries configuration.

**-cpt 5** – The status offload daemon component.

## Configuration File and Command-Line Definitions

During status offloading, the exit initializes the ODBC environment only once. The entry point for the exit is defined and the attributes of the status control records and status detail messages are obtained. The status message is inserted into the database depending on the type of message (based on the XMObject name).

In the FTF configuration file, define the following local MQSeries queues, which are used as input to FTFSTADB:

- StatusControlQueue
- StatusDetailQueue
- StatusUnloadControlQueue

During STATD initialization, three separate states are sent to the exit: initialize, submit, and terminate.

The following stanzas are required in the FTF configuration file for FTFSTATD to run:

- StatusUnloadNumInstances
- StatusUnloadControlQueue
- StatusUnloadDataStore

---

### Note:

When using the FTFStatusPrint entry point, the StatusUnloadDataStore must contain the path and filename of the file where you want to store the status information.

---

## Defining the Configuration File for Status Offload

The FTFSTATD exit name and entry point must be specified in the configuration file. It can be optionally overridden by the FTFSTATD command line.

## Status Offload Daemon

### *Configuration File and Command-Line Definitions*

The StatusUnloadControlQueue must be defined as follows:

```
DEFINE QLOCAL (FTFSTATD.CONTROL) MSGDLVSQ (FIFO) REPLACE
```

The FIFO queue value ensures that the queue can be terminated.

As part of the FTFNodeDefault stanza, the following definitions are required for FTFSTATD in the FTFCONFIG.INI file:

```
StatusControlQueue=FTFSTAT.CONTROL  
StatusDetailQueue=FTFSTAT.DETAIL  
Status=Persist
```

The StatusControlQueue and StatusDetailQueue specify the location of the messages for the FTFSTATD component to retrieve for offloading to a data store. Setting the status to **Persist** allows the status messages to remain on their respective queues during queue manager restarts.

The status message routing can be set so that all messages are routed to a central location. Status information can be correlated from the queue manager when multiple queue managers are involved in the FTF transfer, as follows:

```
StatusServer=queueManager
```

In this sample for unloading status messages, the definitions set in the configuration file are as follows:

```
StatusUnloadNumInstances=1  
StatusUnloadControlQueue=FTFSTATD.CONTROL  
StatusUnloadDLLName=FTFEXSTA  
StatusUnloadDLLEntry=FTFExitStat  
StatusUnloadTimeout=60  
StatusUnloadBatchCount=300  
StatusUnloadDataStore=dataStore  
StatusUnloadUserID=ftfusr  
StatusUnloadUserPassword=password
```

---

**Note:**

The StatusUnloadNumInstances line is commented out in the sample code provided, in order to accommodate OS/390 startup rules. To use the status daemon, you must uncomment this line.

---

### Using the Sample Exit on OS/400

On OS/400, if you choose to use the sample FTFEXSTA exit for FTFSTATD, you must specify a data file in the FTFCONFIG file to hold offloaded status messages, as follows:

```
StatusUnloadDataStore=MYLIB/STATUSLOG
```

Be sure to create a file with adequate record length. The following command, for example, creates a source physical file:

```
CRTSRCPF MYLIB/STATUSLOG RCDLEN(1036)
```

## Defining the Command Line for Status Offload

In the following example, all of the options possible for the FTFSTATD command are specified:

```
FTFSTATD -lqm localQueueMgr [-cfile configFile | -cq configQueueName]  
-ofile optionsFile -lfile logFile -timeout 10 -batchcount 100 -exitdll  
newExSta -exitentry FTFExitStat -exitdata TestData -userid ftfusr -password  
letmein -version [-help | -h | -?]
```

Where:

**-lqm *localQueueMgr*** – Name of the local queue manager that manages the status message queues upon which FTFSTATD operates. If the local queue manager is not specified, the command connects to the default queue manager that is set in the MQSeries configuration.

**-cfile *configFile*** – The fully qualified path and filename for the FTF configuration file. If no cq option is specified on the command line, then cfile must be specified.

## Status Offload Daemon

### *Configuration File and Command-Line Definitions*

**-cq** *configQueueName* – Name of the queue from which the configuration information is to be retrieved for this FTF instance on this node. If no **cfile** option is specified on the command line, then **cq** must be specified. The **cq** option value points to the queue name rather than to the standard configuration file.

**-ofile** *optionsFile* – The fully qualified path and filename of a text file used to contain command-line option values for the FTFSTATD command. In the options file, you can set any of the available FTFSTATD command-line option values. Values specified on the command line override any values in the options file.

**-lfile** *logFile* – Fully qualified path and filename of the log file where the status daemon writes error messages. This value is optional.

**-timeout** – The amount of time in seconds to wait for new data to arrive on an input queue. Data retrieved before the timeout constitutes a single unit of work. This value is optional. **Valid values:** 1 - 84600. **Default value:** 60

**-batchcount** – The maximum number of messages to be included in a single unit of work before passing control to the processing exit. Fewer messages may be included if the timeout value is reached. Note that a high batchcount value might require too much memory and therefore cause an error. This value is optional. **Valid values:** 1 or greater. **Default value:** 300

**-exitdll** – Name of the dynamic load library (DLL) used to process the XML data streams. This value is required and must be specified either in the configuration file or on the command line.

**-exitentry** – Name of the status daemon's entry point in the DLL. This value is required and must be specified either in the configuration file or on the command line.

**-exitdata** – Name of the user-specified text string that is passed to the exit. This value is optional and can be specified only on the command line.

**-userid** – The user name that is passed to the exit and used to check authorization to write files. This value can be specified either on the command line or in the configuration file.

**-password** – The password associated with the user ID.

**-version** – Displays the current version of the status daemon. Any command entered with this option ignores all other option values and returns the version information.

**-help, -h, or -?** – Displays a list and description of the FTFSTATD command-line option values.

## Defining the Command Line for Retrieval from the Sample Relational Database

To retrieve status messages from a relational database that is ODBC-compliant, you can use the following command line syntax with the listed options:

FTFSTADB -datasource *dataSource* -userid *xxx* -password *xxx*

Datasource is required; userid and password are optional. The userid and password can be the same as those for FTFSTATD but are not required to be the same. The userid and password here on the FTFSTADB command line allow the user only read access to the messages stored within the StatusOffload database, if the database has implemented restricted access via user authorizations.

The following option values are filter options that can be used with the FTFSTADB command line:

-format [long | detail | terse] -label (*user defined*) -ftfid *xxx*  
-oqm *originatingQueueMgr* -rqm *replyQueueMgr* -sqm *sourceQueueMgr*  
-dqm *destQueueMgr* -sdate *beginDate* -edate *endDate* -spath  
*sourceFilePath* -dpath *targetFilePath*

## Recommended Database Schema

Two tables are required, a *StatusSummary* table and a *StatusDetail* table. Both tables have a uniquely generated (by DBMS) sequence number for each record.

### Status Summary Table

Int	SeqNo
Char	FTFID
Datetime	FTFTimeStamp
Char	LQM
Char	OQM
Char	SQM
Char	SFilename
Chara	DQM

## Status Offload Daemon

### *Recommended Database Schema*

Int	SeqNo
Char	DFilename
Char	Label
Char	ID1
Char	ID2
Char	ID3
Int	Priority
Int	Persistent
Int	BytesSent
Int	NumMsgs
Int	TransferTime

### Status Detail Table

Int	DetailSeqNo
Int	SeqNo
Char	FTFID
Datetime	FTFTimeStamp
Char	LQM
Int	ComponentID
Int	Status
Char	ErrorText
Int	CurrentMsgNo
Int	TotalMsgCount



---

---

# Tivoli Data Exchange User Exits

This chapter describes how to use user exits to include custom programming in a data-transfer request.

It contains the following sections:

Section	Page
Exploiting User Exit Facilities in FTF	68
FTF User Exit Overview	70
Reporting Status Information from Exit Modules	84
Customizing and Implementing FTF Exits	85

## Assumptions

This chapter makes the following assumptions:

- You understand the roles of the Tivoli Data Exchange (TDE) (FTF) Manager, Sender, and Receiver in the execution of a data-transfer request.
- You understand the C programming language.

## Exploiting User Exit Facilities in FTF

User exits enable you to tightly couple the business-processing functionality with the data-transfer requests. FTF provides several exit points at strategic locations during a data-transfer request. At each of these locations, you can execute customized modules that run as part of the file-transfer request. These modules are called *exit modules* and must be developed in the C language. Components within the FTF architecture provide facilities to invoke the custom functionality.

The user exits are invoked synchronously by the FTF components. The FTF components invoke the user exits in the same thread of execution and do not spawn a separate thread to execute the user exits.

### Connectors

Connectors are FTF user exits that allow you to send and receive data that is not stored in files. They are run from connector modules, programs executed when the connector exits are invoked. .

## Available Exits

The following table lists and describes the user exits currently available in FTF. It lists each user exit's executing node (dqm, oqm, or sqm) and the FTF component (FTF Manager, Sender, or Receiver).

FTF Exit No.	Executing Node	FTF Component	Description
Exit 3	originating queue manager (oqm)	FTF Manager	Manager pre-process exit. If specified, this is the first exit executed in the data-transfer request.
Exit 4	oqm	FTF Manager	Manager post-process. If specified, this is the last exit executed in the data-transfer request.  <b>Note:</b> This exit's execution does not depend on the success or failure of the data-transfer request.
Exit 5	source queue manager (sqm)	FTF Sender	Sender pre-process exit. This exit is invoked before the FTF Sender reads the source file's contents.

**Tivoli Data Exchange User Exits**  
*Exploiting User Exit Facilities in FTF*

FTF Exit No.	Executing Node	FTF Component	Description
Exit 6	sqm	FTF Sender	<p>Sender post-process exit. The FTF Sender invokes this exit after the source file's contents have been read and processed for transfer to the target node.</p> <p><b>Note:</b> This exit is invoked regardless of whether the FTF Sender is able to successfully process the source file.</p>
Exit 7	destination queue manager (dqm)	FTF Receiver	Receiver pre-process exit. The FTF Receiver invokes this exit before it begins to write the target file.
Exit 8	dqm	FTF Receiver	<p>Receiver post-process exit. The FTF Receiver invokes this exit after the target file has been processed and all the source data has been written to it.</p> <p><b>Note:</b> The receiver post-process exit is invoked by the FTF Receiver regardless of whether the target file is processed successfully.</p>
Exit 9	sqm	FTF Sender	Invokes the sender connector exit.
Exit 10	dqm	FTF Receiver	Invokes the receiver connector exit. .

## **FTF User Exit Overview**

FTF uses the dynamic load library (DLL) facilities available on all platforms that support FTF. FTF components dynamically load the user-specified exit and execute the specified functionality.

---

### **Note:**

The 4690 Client platform does not invoke a DLL but submits a command-line request instead. For information on invoking FTF user exits on the 4690 Client, refer to the “Client for 4690” chapter in this *User's Guide*.

---

All user-exit functionality must be compiled into DLLs and the entry points must be exported according to the platform specifications. For more information, see “Compiling and Generating DLLs” on page 87. FTF invokes the entry points with arguments according to the specific exit number.

All exit modules must be written in the C language. The specific data structures for each of the exits are found in the FTFC.H that resides in the Tivoli Data Exchange directory. These data structures are also described in the *Tivoli Data Exchange Technical Reference*.

A parameter specified as input is passed by the calling FTF component to the user exit and is for read-only purposes in the user exit. A parameter specified as output can be used by the user exit to pass information back to the calling FTF component. FTF checks the returned values in the output field and performs appropriate actions. A parameter specified as input/output is used by the FTF component to pass information to the exit. This exit passes information back to the FTF component.

In the AS/400 environment, exits are service programs, rather than DLLs, as they are on other platforms. For information about compiling service programs, see “Compiling and Generating DLLs in AS/400” on page 90.

This section describes the following exits:

- “Manager Pre-Process Exit” (see page 71)
- “Manager Post-Process Exit” (see page 73)
- “Sender Pre-Process Exit” (see page 76)
- “Sender Post-Process Exit” (see page 78)
- “Receiver Pre-Process Exit” (see page 80)
- “Receiver Post-Process Exit” (see page 82)

## Manager Pre-Process Exit

### Exit #

3

### Description

Manager pre-process exit. The FTF Manager executes this pre-process exit after it receives the file-transfer request but before the request is sent to the FTF Sender. If this exit fails, further processing on this data-transfer request is discontinued. An error message with a primary return code of FTFRCE\_MGRPRE\_EXIT\_FAILURE (675) is returned to the calling interface.

### Restrictions

Must be implemented as a DLL on most platforms or as a service program on the AS/400.

### Command-Line Arguments

FTF -exit 3 -exitdll *dllName* -exitEntry *entrypointName* [-exitData *userData*]

Where:

- **-exit** *exitNumber* – Designates the exit being called.
- **-exitdll** *dllName* – The name of the DLL, shared object, load module, or service program that contains the exit module.
- **-exitEntry** *entrypointName* – The name of the function in the DLL to be invoked.
- **-exitData** *userData* – An optional command-line argument where user-specified data may be passed to the exit. If you are running on an OS/390 platform and the jsdataset argument was specified in the FTFMGR command line, FTF enters the supplied data set name in this field.

## **C Prototype**

```
FTFMQDLLEXPORT void entrypointName (FTFExitInfo *pExitInfo,  
FTFExitRequestInfo *pInfo)
```

## **Return Codes**

Void function. Return codes are set within the FTFExitInfo data structure. For more information about the FTFExitInfo data structure, see “FTFExitInfo” in the *Tivoli Data Exchange Technical Reference*.

## **Data Structures**

- FTFExitInfo (see the *Tivoli Data Exchange Technical Reference*, “FTFExitInfo” on page 225)
- FTFExitRequestInfo (see the *Tivoli Data Exchange Technical Reference*, “FTFExitRequestInfo” on page 234)

## **Manager Pre-Process Exit Usage Notes**

The Manager pre-process exit is invoked with all the information about the current data-transfer request. A variety of business processing logic can be implemented using this exit: for example, validation of queue managers involved in the particular transfer, validating source and target file information, generating audit trails, etc.

The rc data element in the FTFExitInfo data structure takes one of the following values:

- 0 (zero) if the data-transfer request is successful to this point
- The primary return code from the first failing component in the data-transfer request

The FTF Manager propagates this value to the end-user as the secondary return code and generates a log message. The FTF Manager checks the rc value before it proceeds with the data-transfer request.

## Manager Post-Process Exit

### Exit #

4

### Description

Manager post-process exit. The FTF Manager executes this exit after the data-transfer request has been completed. This exit is invoked regardless of whether the transfer is successful. Please note that failure can occur in any one of the components involved in the life cycle of the data-transfer request. The manager post-process exit completes the last stage in the unit of work of the data-transfer request.

The FTF Manager returns the return code based on the first reason for the failure of the current data-transfer request. For example, if in a data-transfer request a Manager post-process exit is specified and the FTF Sender failed the request because of the non-availability of the source file, the FTF Sender return code is sent to the calling component even if the post-process exit fails. This enables you to perform the correct diagnosis of the first problem (which, in this case, is the nonavailability of the source file).

In the case of the Manager post-process exit failure, the return code FTFRCE\_MGRPOST\_EXIT\_FAILURE (676) is returned.

### Restrictions

Must be implemented as a DLL on most platforms or as a service program on the AS/400.

### Command-Line Arguments

FTF -exit 4 -exitdll *dllName* -exitEntry *entrypointName* [-exitData *userData*]

Where:

- **-exit** *exitNumber*– Designates the exit being called.
- **-exitdll** *dllName* – The name of the DLL, shared object, load module, or service program that contains the exit module.

- **-exitEntry** *entrypointName* – The name of the function in the DLL to be invoked.
- **-exitData** *userData* – An optional command-line argument where user-specified data may be passed to the exit. If you are running on an OS/390 platform and the jsdataset argument was specified in the FTFMGR command line, FTF enters the supplied data set name in this field.

## C Prototype

```
FTFMQDLLEXPORT void entrypointName (FTFExitInfo *pExitInfo,  
FTFExitRequestInfo *pInfo)
```

## Return Codes

Void function. Return codes are set within the FTFExitInfo data structure. For more information about the FTFExitInfo data structure, see “FTFExitInfo” in the *Tivoli Data Exchange Technical Reference*.

## Data Structures

- FTFExitInfo (see the *Tivoli Data Exchange Technical Reference*, “FTFExitInfo” on page 225)
- FTFExitRequestInfo (see the *Tivoli Data Exchange Technical Reference*, “FTFRequestMsgInfo” on page 256)

## Manager Post-Process Exit Usage Notes

The Manager post-process exit is invoked with all the information about the current data-transfer request. Because this exit signifies the end of the unit of work, a number of business and user-specific logic can be implemented including updating of audit trails, customized status reporting, and interfacing with any of your systems.

The rc data element in the FTFExitInfo data structure takes one of the following values:

- 0 (zero) if the data-transfer request is successful to this point



- The primary return code from the first failing component in the data-transfer request

The exit checks this data element and responds according to the return code. The exit sets the rc data element to 0 to indicate success in the exit, or to a nonzero value, to indicate failure in the exit. The FTF Manager checks this field for a successful return code.

## **Sender Pre-Process Exit**

### **Exit #**

5

### **Description**

Sender pre-process exit. The FTF Sender executes this exit before reading the source file. If this exit fails, further processing on this data-transfer request is discontinued. An error message with a primary return code of FTFRCE\_SDRPRE\_EXIT\_FAILURE (677) is returned to the calling interface.

### **Restrictions**

Must be implemented as a DLL on most platforms or as a service program on the AS/400.

### **Command-Line Arguments**

FTF -exit 5 -exitdll *dllName* -exitEntry *entrypointName* [-exitData *userData*]

Where:

- **-exit** *exitNumber*– Designates the exit being called.
- **-exitdll** *dllName* – The name of the DLL, shared object, load module, or service program that contains the exit module.
- **-exitEntry** *entrypointName* – The name of the function in the DLL to be invoked.
- **-exitData** *userData* – An optional command-line argument where user-specified data may be passed to the exit. If you are running on an OS/390 platform and the jsdataset argument was specified in the FTFMGR command line, FTF enters the supplied data set name in this field.

## **C Prototype**

```
FTFMQDLLEXPORT void entrypointName (FTFExitInfo *pExitInfo,  
FTFExitSourceFileInfo *pInfo)
```

## **Return Codes**

Void function. Return codes are set within the FTFExitInfo data structure. For more information about the FTFExitInfo data structure, see “FTFExitInfo” in the *Tivoli Data Exchange Technical Reference*.

## **Data Structures**

- FTFExitInfo (see the *Tivoli Data Exchange Technical Reference*, “FTFExitInfo” on page 225)
- FTFExitSourceFileInfo (see the *Tivoli Data Exchange Technical Reference*, “FTFExitSourceFileInfo” on page 236)

## **Sender Pre-Process Exit Usage Notes**

The Sender pre-process exit is invoked with information about the exit (stored in the FTFExitInfo data structure), and information about the source file being transferred (stored in the FTFExitSourceFileInfo data structure). A variety of business processing logic can be implemented using this exit, such as validating source file information, security validations, generating audit trails, running a business process which creates the source file, etc.

The rc data element in the FTFExitInfo data structure takes one of the following values:

- 0 (zero) if the data-transfer request is successful to this point
- The primary return code from the first failing component in the data-transfer request

The FTF Sender propagates this data element value to the end-user as the secondary return code and generates a log message. The rc data element must be set to 0 to indicate success in the exit. The FTF Sender checks the rc value before proceeding with the data-transfer request.

## **Sender Post-Process Exit**

### **Exit #**

6

### **Description**

Sender post-process exit. The FTF Sender executes this exit after reading the source file and depositing the data messages on the outbound target queues. If this exit fails, further processing on this data-transfer request is discontinued. An error message with a primary return code of 678, FTFRCF\_SDRPOST\_EXIT\_FAILURE, is returned to the calling interface.

### **Restrictions**

Must be implemented as a DLL on most platforms or as a service program on the AS/400.

### **Command-Line Arguments**

FTF -exit 6 -exitdll *dllName* -exitEntry *entrypointName* [-exitData *userData*]

Where:

- **-exit** *exitNumber* – Designates the exit being called.
- **-exitdll** *dllName* – The name of the DLL, shared object, load module, or service program that contains the exit module.
- **-exitEntry** *entrypointName* – The name of the function in the DLL to be invoked.
- **-exitData** *userData* – An optional command-line argument where user-specified data may be passed to the exit. If you are running on an OS/390 platform and the jsdataset argument was specified in the FTFMGR command line, FTF enters the supplied data set name in this field.

## **C Prototype**

```
FTFMQDLLEXPORT void entrypointName (FTFExitInfo *pExitInfo,  
FTFExitSourceFileInfo *pInfo)
```

## **Return Codes**

Void function. Return codes are set within the FTFExitInfo data structure. For more information about the FTFExitInfo data structure, see “FTFExitInfo” in the *Tivoli Data Exchange Technical Reference*.

## **Data Structures**

- FTFExitInfo (see the *rTivoli Data Exchange Technical Reference*, “FTFExitInfo” on page 225)
- FTFExitSourceFileInfo (see the *Tivoli Data Exchange Technical Reference*, “FTFExitSourceFileInfo” on page 236)

## **Sender Post-Process Exit Usage Notes**

The Sender post-process exit is invoked with information about the exit (stored in the FTFExitInfo data structure) and information about the source file being transferred (stored in the FTFExitSourceFileInfo data structure). The FTF Sender executes this exit whether or not the source file is successfully processed. A variety of business processing logic can be implemented using this exit, such as validating source file information, security validations, generating audit trails, and running a business process which deletes the source file to prepare for the next transfer.

The rc data element in the FTFExitInfo data structure takes one of the following values:

- 0 (zero) if the data-transfer request is successful to this point
- The primary return code from the first failing component in the data-transfer request

The exit checks this field and responds according to the return code. The exit sets this field to 0 to indicate success in the exit or to a nonzero value to indicate failure in the exit. The FTF Sender checks this field for a successful return code.

## Receiver Pre-Process Exit

### Exit #

7

### Description

Receiver pre-process exit. The FTF Receiver executes this exit before processing the data messages and writing to the target file. If this exit fails, further processing on this data-transfer request is discontinued. An error message with a primary return code of FTFRCE\_RCVPRE\_EXIT\_FAILURE (679) is returned to the calling interface.

### Restrictions

Must be implemented as a DLL on most platforms or as a service program on the AS/400.

### Command-Line Arguments

FTF -exit 7 -exitdll *dllName* -exitentry *entrypointName* [-exitdata *userData*]

Where:

- **-exit** *exitNumber* – Designates the exit being called.
- **-exitdll** *dllName* – The name of the DLL, shared object, load module, or service program that contains the exit module.
- **-exitEntry** *entrypointName* – The name of the function in the DLL to be invoked.
- **-exitData** *userData* – An optional command-line argument where user-specified data may be passed to the exit. If you are running on an OS/390 platform and the jsdataset argument was specified in the FTFMGR command line, FTF enters the supplied data set name in this field.

## **C Prototype**

```
FTFMQDLLEXPORT void entrypointName (FTFExitInfo *pExitInfo,  
FTFExitTargetFileInfo *pInfo)
```

## **Return Codes**

Void function. Return codes are set within the FTFExitInfo data structure. For more information about the FTFExitInfo data structure, see “FTFExitInfo” in the *Tivoli Data Exchange Technical Reference*.

## **Data Structures**

- FTFExitInfo (see the *Tivoli Data Exchange Technical Reference*, “FTFExitInfo” on page 225)
- FTFExitTargetFileInfo (see the *Tivoli Data Exchange Technical Reference*, “FTFExitTargetFileInfo” on page 241)

## **Receiver Pre-Process Exit Usage Notes**

The Receiver pre-process exit is invoked with information about the exit (stored in the FTFExitInfo data structure) and information about the target file that must be created (stored in the FTFExitTargetFileInfo data structure). A variety of business processing logic can be implemented using this exit, such as validating target file information, security validations, and generating audit trails.

The rc data element in the FTFExitInfo data structure takes one of the following values:

- 0 (zero) if the data-transfer request is successful to this point
- The primary return code from the first failing component in the data-transfer request

The FTF Receiver propagates this to the end-user as the secondary return code and generates a log message. The rc value must be set to 0 to indicate success in the exit. The FTF Receiver checks the rc value before it proceeds with the data-transfer request.

## **Receiver Post-Process Exit**

### **Exit #**

8

### **Description**

Receiver post-process exit. The FTF Receiver executes this exit after processing the target file. This exit is invoked by the FTF Receiver regardless of whether the target file is successfully processed. If this exit fails, an error message with a primary return code of FTFRCE\_RCVPOST\_EXIT\_FAILURE (680) is returned to the calling interface.

### **Restrictions**

Must be implemented as a DLL on most platforms or as a service program on the AS/400.

### **Command-Line Arguments**

FTF -exit 8 -exitdll *dllName* -exitentry *entrypointName* [-exitdata *userData*]

Where:

- **-exit** *exitNumber*– Designates the exit being called.
- **-exitdll** *dllName* – The name of the DLL, shared object, load module, or service program that contains the exit module.
- **-exitEntry** *entrypointName* – The name of the function in the DLL to be invoked.
- **-exitData** *userData* – An optional command-line argument where user-specified data may be passed to the exit. If you are running on an OS/390 platform and the jsdataset argument was specified in the FTFMGR command line, FTF enters the supplied data set name in this field.



## **C Prototype**

```
FTFMQDLLEXPORT void entrypointName (FTFExitInfo *pExitInfo,  
FTFExitTargetFileInfo *pInfo)
```

## **Exit Return Codes**

Void function. Return codes are set within the FTFExitInfo data structure. For more information about the FTFExitInfo data structure, see “FTFExitInfo” in the *Tivoli Data Exchange Technical Reference*.

## **Data Structures**

- FTFExitInfo (see the *Tivoli Data Exchange Technical Reference*, “FTFExitInfo” on page 225)
- FTFExitTargetFileInfo (see the *Tivoli Data Exchange Technical Reference*, “FTFExitTargetFileInfo” on page 241)

## **Receiver Post-Process Exit Usage Notes**

The Receiver post-process exit is invoked with information about the exit (stored in the FTFExitInfo data structure) and information about the source file being transferred (stored in the FTFExitTargetFileInfo data structure). A variety of business processing logic can be implemented using this exit, such as validating target file information, security validations, generating audit trails, and running a business process which processes the target file and applies the changes to a database.

The rc data element in the FTFExitInfo data structure takes one of the following values:

- 0 (zero) if the data-transfer request is successful to this point
- The primary return code from the first failing component in the data-transfer request

The exit checks this value and takes necessary actions based on the return code. The exit sets this field to 0 to indicate success in the exit and to a nonzero value for failure in the exit. The FTF Receiver checks this field for a successful return code.

## Reporting Status Information from Exit Modules

FTF allows you to send custom status messages to the FTF status subsystem. The status subsystem treats the status information from your exit module the same as any other status information it processes. This extension of the status subsystem allows you to submit any type of information required for your custom processing.

To submit custom status information, follow these steps:

1. In your exit module, define the following variables:
  - custom component
  - custom status type
  - error text (optional)

You can put any data into these three data items. Each data item is processed as character data. The data you insert in these data items is not required to conform with internal FTF status items.

2. In the exit module, call the `FTFSubmitStatusMsg()` function, passing the custom component, custom status type, and error text. This function has the following syntax:

```
FTFVOID FTFSubmitStatusMsg
      (FTFExitInfo  *pFTFExitInfo,
      FTFCHAR      *customComponent,
      FTFCHAR      *customStatusType,
      FTFCHAR      *errorText,
      FTFCA        *pftfca);
```

Where:

- **pFTFExitInfo** – Input parameter that was passed into the exit. For more information about the “FTFExitInfo” on page 225 in the *Tivoli Data Exchange Technical Reference*.
- **customComponent** – Input parameter that contains the custom component value set in your exit module.
- **customStatusType** – Input parameter that contains the custom status type value set in your exit module.
- **errorText** – Input parameter that contains the error text set in your exit module.

- **\*pftfca** – Input parameter that contains the FTFCA data structure, which contains exit return code information. For more information about the FTFCA data structure, see “FTFCA” in the *Tivoli Data Exchange Technical Reference*.

## Example

In this example, the exit module loads a flat file into a database. Custom component, custom status, and error text values are set to indicate the operation failed, and the message information is submitted to the status subsystem.

```
...
FTFCHAR *custComp="Update Database";
FTFCHAR *custStat="Update Failed";
FTFCHAR *errorText="Database insert failed";
FTFSubmitStatusMsg(pExitInfo, custComp, custStat,
    errorText, pFtfca);
```

## Customizing and Implementing FTF Exits

Customizing and implementing FTF exits involves the following tasks:

- Changing the sample exits to suit your needs (adding business logic) (see page 86)
- Compiling and generating DLLs on different platforms (see page 87)
- Making the Exits available (see page 90)

## Sample Exits

FTF is shipped with the following sample exit modules both in the form of executable binaries and source code so you can use them as the basis for implementing your exit modules.

Name	Description	OS/390	Function	
			UNIX	Windows NT
FTFEX34	Sample exit for Manager pre- and post-processing. Provides entry points for the Manager pre- and post-processing exits. <b>FTFMgrPre3</b> – Manager pre-process exit entry point. <b>FTFMgrPost4</b> – Manager post-process exit entry point.	Prints the data specified in the exitdata argument to standard output.	Executes the system function call with the value specified in the exitdata argument.	Executes the system function call with the value specified in the exitdata argument with the Windows NT start command prefixed.
FTFEX56	Sample exit for Sender pre- and post-processing. Provides entry points for the Sender pre- and post-processing exits. <b>FTFSdrPre5</b> – Sender pre-process exit entry point. <b>FTFSdrPost6</b> – Sender post-process exit entry point.	Prints the data specified in the exitdata argument to standard output.	Executes the system function call with the value specified in the exitdata argument.	Executes the system function call with the value specified in the exitdata argument with the Windows NT start command prefixed.
FTFEX78	Sample exit for Receiver pre- and post-processing. Provides entry points for the Receiver pre- and post-processing exits. <b>FTFRcvPre7</b> – Receiver pre-process exit entry point. <b>FTFRcvPost8</b> – Receiver post-process exit entry point.	Prints the data specified in the exitdata argument to standard output.  <b>Note:</b> With an SMP/E installation, the name is <b>FTFPEX78</b> .	Executes the system function call with the value specified in the exitdata argument.	Executes the system function call with the value specified in the exitdata argument with the Windows NT start command prefixed.

## Compiling and Generating DLLs

Compiling and generating DLLs is a platform-specific task. Each platform has its own nuances and procedures for creating DLLs. This section provides a general overview for each of the FTF supported platforms. It is not intended to be a substitute for the platform's documentation. Refer to the specific platform's documentation for more information on how to create DLLs.

The examples in this section are for customizing the manager pre-process exit. The Manager pre-process prototype is as follows:

```
FTFMQDLLEXPORT void entryPointName (FTFExitInfo *pExitInfo,  
FTFExitRequestInfo *pInfo)
```

Where *entryPointName* is the name of the exit module function.

You can copy the supplied source module FTFEX34.C and customized for your needs based on the following guidelines.

### Compiling and Generating DLLs in OS/390

1. OS/390 has no special prefix that must be applied to the function definition to export its definition. Use the following definition as an empty definition for FTFMQDLLEXPORT.

```
#define FTFMQDLLEXPORT
```

This step is already included in the sample source code.

2. Explicit program declarations must be made to export each entry point in a DLL. Specify a definition, as shown below, for each entry point that you wish to include in the DLL.

```
#pragma export entryPointName
```

Where *entryPointName* is the name of the exit module function.

3. Implement the desired functionality within the entry point by following the guidelines in the sample code. Note that the rc data element in the FTFExitInfo data structure is both input and output and must be properly set to indicate success or failure.

## Tivoli Data Exchange User Exits

### Customizing and Implementing FTF Exits

4. Compile the modified source code using the following JCL as a guideline. Refer to the C Programming Guide for OS/390 for more information about compiling DLLs.

```
//JOB CARD
//COMP1      EXEC  EDCC,CPARM='LONGNAME RENT EXPORTALL DLL',
//              INFILE=++SOURCEDATASET++ (++MEMBER++) ,
//              OUTFILE=++OBJDATASET++ (++MEMBER++)
//LINK1      EXEC  EDCPL,PPARM='NONCAL NOMAP',
//              OUTFILE=++LOADDATASET++ (++MEMBER++) ,DISP=SHR
//PLKED.SYSIN DD  DSN=++OBJECTDATASET++ (++MEMBER++) ,DISP=SHR
//PLKED.SYSMOD DD  DSN=++PLOBJECTDATASET++ (++MEMBER++) ,DISP=SHR
//PLKED.SYSDEFSD DD DSN=++SIDEDECKDATASET++ (++MEMBER++) ,DISP=SHR
//
```

---

### Note:

You must specify the DLL compiler option and the side deck data set and must customize this JCL to specify the include search path for the FTF include files.

---

## Compiling and Generating DLLs in Windows NT

### Microsoft Developer Studio Procedure

1. Use the project wizard on the Developer Studio workbench to create a new project for Win32 Dynamic-Link Library.
2. Use the Files wizard to open a new file called *yourDll.c*.
3. You can either copy an exit source code supplied with FTF and modify it or build the file from scratch.
4. Build business functionality in the exit entry function.
5. The functions are exported on Windows NT by specifying `_declspec(dllexport)` as part of the function definition. Use the following definition as an empty definition for `FTFMQDLLEXPORT`.

```
#define FTFMQDLLEXPORT _declspec(dllexport)
```

This step is already included in the sample.

6. On the Project Settings, specify the FTF include directories where `FTFC.H` can be found.

7. If necessary, change the DLL name.
8. Build the project and generate the DLL.

## **Compiling and Generating DLLs in UNIX**

1. On UNIX operating systems, no special prefix needs to be applied to the function definition to export its definition. Use the following definition as an empty definition for FTFMQDLLEXPORT.

```
#define FTFMQDLLEXPORT
```

This step is already included in the sample source code.

2. Implement your functionality within the entry point by following the guidelines in the sample code. Remember that the rc data element in the FTFExitInfo data structure is both input and output and must be set properly to indicate success or failure.
3. Compile your modified source code and link using the following linker share options where exports is the filename that holds the names of your exit entry points.

### **AIX Options**

```
-bE:exports -bM:SRE -H512 -T512 -e ftfentry
```

For example, the sample code exports file will hold the following:

```
ftfentry
```

```
FTFMgrPre3
```

```
FTFMgrPost4
```

### **HPUX Options**

```
cc -Aa +z -c FTFEX34.c
```

```
d -b -o FTFEX34.sl FTFEX34.o
```

### **Sun Solaris Options**

-G -e ftfentry

---

**Note:**

An export file is not required on HP-UX and Sun Solaris platforms.

---

## **Compiling and Generating DLLs in AS/400**

1. Use the CRTCMOD command to compile the exit source code into a module.
2. After a successful compile, use the CRTSRVPGM command to create the service program. Specify the EXPORT parameter.

## **Making Exits Available**

After the DLL is built, it must be made available for dynamic loading on each of the platforms. The procedure varies according to platform.

### **Making Exits Available in OS/390**

The DLL or load module must be copied to a data set which is part of the STEPLIB of the FTFSTART process.

Please note that the FTFSTART process abends with a MODULE NOT FOUND error if you specify a DLL in the command line which is not available in the STEPLIB.

### **Making Exits Available in Windows NT**

The DLL should be made available in a directory that is included in the PATH variable. Note that after the FTF components have been started, changing the PATH variable has no effect. After changing the path variable, you must restart the FTF components.



## **Making Exits Available in UNIX**

Choose a directory to hold the DLL(s) and copy the DLL(s) to this directory. Set the environment variable FTF\_EXIT\_PATH to this directory.

## **Making Exits Available in AS/400**

Choose a library to hold the service program and copy the service program to the library. Add the library to the library list so that it can be found when called.

## **Protected Job Submission Exit for OS/390**

You can specify a predefined data set on the FTFMGR, FTFSDR, and FTFRCV command lines using a new argument, jsdataset, which restricts your job to a predefined job stream. This argument is valid only on OS/390. (For more information about the jsdataset argument, see “Component Configuration Commands” in the *Tivoli Data Exchange Technical Reference*.)

To use this exit, you must specify the jsdataset argument to start FTF components. When you submit a job, you specify the name of the data set member, which contains the JCL. If you submit a job that is not included in the startup data set, the job is not accepted. This protects the system from unwanted jobs.

To start FTF components with the jsdataset specified, edit the FTFJSTRT JCL. The prototype is as follows:

```
-exit X -exitdll FTFEXJSP -exitentry FTFSUBJ -exitdata "MEMBER1,  
MEMBER2"
```

Where:

- **exit** is exit 3 through 8.
- **MEMBER1** is the JCL to be submitted if the transfer is successful.
- **MEMBER2** is the JCL to be submitted if the transfer fails.

---

### **Note:**

MEMBER2 is optional. If you specify only MEMBER1, it is executed only if the transfer is successful.

---

## Tivoli Data Exchange User Exits

### *Customizing and Implementing FTF Exits*

Sample JCL for the edited FTFJSTRT follows.

```
//FTFSTART PROC QMGR=MQA2,
//          FTFHLQ='FTFV2.TDE',
//          MQMHLQ='MQM.V1R2M0',
//          INIFILE='FTFV2.TDE.TDE.INI',
//          JSDTASET='FTFV2.JCL.CNTL',
//          MEM='0M',
//          OUTCLASS='*'
//STEP1     EXEC PGM=FTFSTART,
//          REGION=&MEM,
//          PARM='-LQM &QMGR -JSDATASET &JSDTASET -CFILE DD:FTFINI'
//STEPLIB   DD DSN=&FTFHLQ..TDE.LOADLIB,DISP=SHR
//          DD DSN=&MQMHLQ..SCSQAUTH,DISP=SHR
//          DD DSN=&MQMHLQ..SCSQANLE,DISP=SHR
//FTFINI    DD DSN=&INIFILE,DISP=SHR
//SYSOUT    DD SYSOUT=&OUTCLASS
//SYSPRINT  DD SYSOUT=&OUTCLASS
//SYSERR    DD SYSOUT=&OUTCLASS
//SYSUDUMP  DD SYSOUT=&OUTCLASS
//FTFSTART  PEND
//*
//FTFSTART  EXEC FTFSTART
```

In this example, the JCL located at FTFV2.JCL.CNTL(FTFGOOD) is submitted to the FTF Receiver.

The following general restrictions apply to this argument:

- Member name on OS/390 cannot exceed 8 characters.
- Fully qualified PDS name (MEMBER) cannot exceed 44 characters.
- Validation at startup does not include jsdataset or exitdata. Therefore, if the argument passed to jsdataset or the member name passed to exitdata does not exist, then the exit produces an error message and fails when executed.

---

# Security Authorization Exits

Tivoli Data Exchange(TDE) includes file-authorization exits on the TDE Sender and the TDE Receiver. These exits allow you to implement and call authorization modules, which can contain customized authorization logic you need for secure data transfers. Depending on the platform, the authorization module is called as a DLL, a shared library, a load module, or a service program in a library list.

The authorization exits are activated by startup parameters. You can specify these parameters in the FTFSTART module in OS/390 or in the Sender and Receiver startup commands in other platforms.

This chapter describes how to use security authorization exits and lists technical reference information about the exits and the data structures they use. It contains the following information:

Section	Page
Customizing the Authorization Module	94
Installing the Security Authorization Exit	96
Interface for Customization	100
Security Authorization Exit Reference	102

## Assumptions

This section makes the following assumptions:

- You have successfully installed MQSeries.
- You have successfully installed TDE on the same machine.

## Customizing the Authorization Module

The authorization module contains the code that uses the security information provided by TDE's security authorization exit. You can use security authorization exits to perform a wide range of functions, from creating an audit trail to making sure the current user has the privileges to perform a specific data transfer.

TDE includes an authorization module for OS/390 installations. This authorization module validates the TDE user who issued the data-transfer request. You can customize this authorization module or build your own authorization modules for any supported operating system. You can find the OS/390 authorization module in the FTF ++FTFHLQ++.SAMPLIB data set on OS/390, or in the file named FTFAUTH.C in the TDE directory on other platforms.

In customizing an authorization module, you should consider the following tasks:

- Using the data passed to the authorization module
- Customizing the return codes

## Using Authorization Module Data Structures

TDE uses two data structures with authorization modules. The FTFExitAuthInfo data structure contains the following security-related information:

- UserID – System user ID for the person performing the transfer
- Filename – Name of the file being transferred
- Windows NT Domain Name – For NT machines only; otherwise, this value is null
- Access Type – *Read* on the TDE Sender, *write* on the TDE Receiver

The data elements in this data structure are used for input to the security authorization module. Use them as necessary to create a custom authorization module. For more information about this data structure see the *Tivoli Data Exchange Technical Reference*, “FTFExitAuthInfo” on page 219 and “FTFExitInfo” on page 225.

The FTExitInfo data structure describes the authorization module's attributes. The information in this structure is used to find and execute the authorization module. It also allows you to identify the status of the custom exit. The status information is an output return code contained in the rc data element in the FTExitInfo data structure. For more information about this data structure see the *Tivoli Data Exchange Technical Reference*, "FTExitInfo" on page 225.

## Authorization Module Return Codes

TDE sets the authorization module's primary return codes. If the security authorization exit fails, it generates the following primary return codes:

<b>TDE Component</b>	<b>Return Code</b>	<b>Description</b>
TDE Sender	FTFRCE_SDRAUTH_EXIT_FAILURE (681)	TDE Sender authorization exit failure
TDE Receiver	FTFRCE_RCVAUTH_EXIT_FAILURE (682)	TDE Receiver authorization exit failure
Both	FTFRCE_EXITLOAD_FAILED (655)	Unable to load exit

TDE also allows you to set secondary return codes that you can define in your authorization module. These secondary codes can provide more information than the primary codes and that information can be customized to meet your processing and business needs.

## Return Codes in the Supplied OS/390 Authorization Module

The OS/390 authorization module generates the following secondary return codes:

<b>Return Code</b>	<b>Description</b>
-4	User not authorized
>4	RACF internal error
76	Invalid resource type
80	Invalid user ID (User IDs must be eight characters and can contain only letters [A-Z] or numbers [0-9].)
84	RACF verify failed
88	Invalid parm list length
92	Caller is not APF authorized
96	Invalid parm list from caller

## **Using Authorization Module Return Codes**

You can use the primary return codes returned by the authorization module in combination with the secondary return codes set in the authorization module code to create a detailed picture of any errors that might occur in an authorization module. These return codes are returned in the FTFFExitInfo data structure's rc data element.

For instance, the OS/390 security authorization module run on OS/390 returns an rc of 681 -4. The primary return code (681) indicates that the authorization error occurred on the Sender. The secondary return code (-4) indicates that the user is not authorized.

If you create your own security authorization module, you can define your own secondary return codes, based on your business needs.

## **Installing the Security Authorization Exit**

Before you can use the authorization module, you must perform the following steps:

1. Relink the TDE components as APF authorized (OS/390 only).
2. Restart them with appropriate security startup parameters. You can use the following methods to restart the TDE components:
  - In OS/390, use the FTFFSTART module to restart them collectively. (See page 97.)
  - In Windows NT, restart the TDE service. (See page 98.)
  - On any platform, restart the components individually. (See page 99.)

## Using FTFSTART on OS/390

The FTFSTART command restarts the TDE Manager, Senders, and Receivers under one address. FTFSTART also enables the security authorization exits.

To restart components collectively using the FTFSTART module, follow these steps:

1. Use the FTFLINK JCL (see “FTFSTART Relink JCL” in the *Tivoli Data Exchange Technical Reference*) to relink FTFSTART as APF authorized. Make the entire FTF LOADLIB APF authorized.
2. Enter the following command to restart TDE components:

```
ftfstart -sdrauth -exitdll sDllName -exitentry  
         sEntryPoint -rcvauth -exitdll rDllName  
         -exitentry rEntryPoint
```

Where:

- *sDllName* – Name of the DLL that contains the TDE Sender’s authorization module.
- *sEntryPoint* – The TDE Sender’s entry point to the DLL. On OS/390 platforms, this value cannot be longer than eight characters.
- *rDllName* – Name of the DLL that contains the TDE Receiver’s authorization module.
- *rEntryPoint* – The TDE Receiver’s entry point to the DLL. On OS/390 platforms, this value cannot be longer than eight characters.

## Security Authorization Exits

### *Installing the Security Authorization Exit*

The sdrauth parameter enables authorization checking in the TDE Sender. FTFSTART uses the FTFAUTH command-line option to start the TDE Sender. The revauth parameter enables authorization checking in the TDE Receiver. FTFSTART uses the FTFAUTH command-line option to start the TDE Receiver.

### Using an Options File

If you use JCL or a TSO command line to start the FTFSTART command, you cannot pass more than 100 characters in the PARM field. To alleviate this problem, you can use the ofile option to pass the FTFSTART arguments.

To use the ofile option, follow these steps:

1. Select a physical sequential file or a PDS member to hold all command-line options.
2. Edit the file to enter the command-line options.
3. Make sure that options do not span lines.
4. Use the unnum command in the ISPF edit session to make sure that you set the edit profile for the data set or member to be unnumbered.
5. For the PARM field in the FTFSTART EXEC statement, specify PARM='-ofile *ofileName*' where *ofileName* is the name of the options file.

Your options filename can be either the fully qualified data set name without quotes or a DD specified in the JCL. The JCL listed in “FTFSTART Execution JCL” in the *Tivoli Data Exchange Technical Reference* uses the options file.

## Restarting the TDE Service

To install security authorization exits in the Windows NT environment, modify the TDE initialization file to include the ftfauth parameters for all TDE Senders and Receivers.

The following example displays the environment section for an instance in which the security authorization exits are installed. In this example, one TDE Sender and one TDE Receiver exist and the sample authorization module is installed on both.



---

**Note:**

You must modify the sample authorization module before you can use it in the Windows NT environment.

---

```
production:
lqm = PRODMQM
cfile = c:\ftfbin\ftfconfig.ini
manager.commandline      = -lfile c:\ftf\mgr.out
sender.commandline.1     = -ftfauth -exitdll FTFAUTH
                        -exitentry FTFauthex1
receiver.commandline.1   = -ftfauth -exitdll FTFAUTH
                        -exitentry FTFauthex1
```

## Restarting Components Individually

On any supported platform, you can restart the affected TDE components individually. Use the appropriate command's `ftfauth` parameter to install the security authorization modules.

To restart the TDE components and install the user authorization module, perform the following steps:

1. Relink the TDE Sender and the TDE Receiver as APF authorized. Make the entire FTF LOADLIB APF authorized. (OS/390 only)
2. Enter the following command to install the security exit on the TDE Sender:

```
ftfsdr -ftfauth -exitdll dllName -exitentry
entryPoint
```

Where:

- *dllName* – Name of the DLL that contains the TDE Sender's authorization module.
  - *entryPoint* – The DLL's entry point.
3. Enter the following command to install the security exit on the TDE Receiver:

## Security Authorization Exits

### *Interface for Customization*

```
ftfrcv -ftfauth -exitdll dllName -exitentry  
      entryPoint
```

Where:

- *dllName* – Name of the DLL that contains the authorization module.
- *entryPoint* – The DLL's entry point.

## Interface for Customization

### Prototype

```
DLLEXPORT null entryPoint(FTFExitInfo *pExitInfo, FTFExitAuthInfo  
*pInfo);
```

### Description

This function is the authorization module that provides security authorization. You can use any name you want for it, but the name must agree with the *exitentry* parameter in the command used to invoke the security authorization exit. In OS/390 environments, the entry point name cannot be longer than eight characters.

### Return Value

None

### Parameters

- **\*pExitInfo** – Address of the variable that contains the FTFExitInfo data structure associated with the authorization module.
- **\*pInfo** – Address of the variable that contains the FTFExitAuthInfo data structure associated with the authorization module.

## Related Data Structures

- FTFExitAuthInfo (see the *Tivoli Data Exchange Technical Reference*, “FTFExitAuthInfo”)
- FTFExitInfo (see the *Tivoli Data Exchange Technical Reference*, “FTFExitInfo”)

## Sample Shell

The following sample is a shell for a security authorization module called FTFRecExitAuth. As indicated previously, you can choose the appropriate name for your authorization module.

In this shell, the \*pExitData value is set to null and the return code is set to 0. When you insert custom code for your authorization module, modify these values as necessary.

```
DLLEXPORT void FTFRecExitAuth(FTFExitInfo *pExit-
Info, FTFExitAuthInfo *pInfo)
{
    char *pExitData=NULL;
    pExitInfo->rc=0;

    /* insert authorization module logic*/

    return;
}
```

# Security Authorization Exit Reference

## Description

The security authorization exit reference allows you to plug in a custom-built authorization module for securing TDE transfers. Separate security authorization exits exist for the TDE Sender and Receiver.

## Invoked From

- TDE Sender
- TDE Receiver

## Command Lines

The following command line invokes the security authorization exit from the sender and receiver:

```
ftfsdr -ftfauth -exitdll dllName -exitentry  
      -entryPoint  
ftfrcv -ftfauth -exitdll dllName -exitentry  
      -entryPoint
```

Where:

- *dllName* – Name of the DLL, shared library, service program, or load module that contains the authorization module.
- *entryPoint* – Name of the function to run as the authorization module. This function must exist within the DLL.

## Associated Data Structures

- FTExitAuthInfo (see the *Tivoli Data Exchange Technical Reference*, “FTExitAuthInfo”)
- FTExitInfo (see the *Tivoli Data Exchange Technical Reference*, “FTExitInfo”)

## **OS/390 Authorization Exit**

TDE includes a working authorization module for OS/390. It is supported as part of the base product. The code for this module checks the user name, file, and access type contained in the FTExitAuthInfo data structure against the external security manager via SAF.

## **Sample Exits**

TDE includes sample code that shows how to use authorization exits. This code applies to all supported platforms and is located in the *Samples* directory in the FTFAUTH.C file.

## **Usage Notes**

For the Sender and Receiver to successfully use security authorization exits, the entire FTFV2 LOADLIB must be APF-authorized.



---

# Working with AS/400 Files

---

This version of Tivoli Data Exchange(TDE) allows you to send and receive data that resides in AS/400 physical and logical files. This chapter describes that functionality and lists command-line examples for running data-transfer requests involving AS/400 physical and logical files.

It contains the following sections:

Section	Page
Overview	106
Modifying the FTF Configuration File	107
Using the FTF Command	108
Examples	111
Determining the Transfer's Success	121

## Assumptions

This chapter makes the following assumptions:

- You have experience with AS/400 physical and logical files and you understand how they are used.
- You understand TDE (FTF) data-transfer capabilities.

## Overview

TDE (FTF) includes the ability to transfer data to and from AS/400 physical and logical files. This ability not only extends the basic data-transfer capabilities of FTF, it allows you to use the FTF product to convert data in text format to and from AS/400 physical and logical file formats. You can use multiple queue managers. Each instance of FTF must be installed against a single queue manager, but you can install multiple instances of FTF against different queue managers.

---

### Note:

The ability to handle AS/400 physical and logical data has the following limitations:

- Data type conversion is somewhat flexible. For example, char and varchar data are not considered as distinct data types.
  - On the FTF Receiver, FTF can only write to the first record format of a logical file. As a result, you should consider using logical files with a single record format.
- 

To send or receive AS/400 physical or logical data, follow these steps:

1. Modify the FTF configuration file's FileTypes property.
2. Execute the transfer. This chapter describes how to use the FTF command to execute transfers with AS/400 physical or logical data. (For more information, see "Using the FTF Command" on page 108.) For information about how to use other interfaces, see the following chapters in this User's Guide:
  - "Using the ISPF User Interface"
  - "Using the 5250 User Interface"
  - "Using the FTF GUI"
  - *Tivoli Data Exchange Technical Reference*, "C API Reference"
  - *Tivoli Data Exchange Technical Reference*, "COBOL API Reference"



3. If necessary, check the FTF status queues to determine whether the transfer executed successfully. (For more information, see “Determining the Transfer’s Success” on page 121.) This chapter describes how to use the FTFSTAT command to check status. For information about how to use other interfaces, see the following chapters in this User’s Guide:
  - “Using the ISPF User Interface”
  - “Using the 5250 User Interface”
  - “Using the FTF GUI”
  - *Tivoli Data Exchange Technical Reference*, “C API Reference”
  - *Tivoli Data Exchange Technical Reference*, “COBOL API Reference”

## Modifying the FTF Configuration File

Before you can send or receive AS/400 physical or logical file data, you must modify the FTF configuration file to define the file type value that is required for the transfer.

To modify the FTF configuration file, open it with a text editor, modify it, then save the results. For complete information about the FTF configuration file, see “FTF Configuration” in the *Tivoli Data Exchange Installation Guide*.

The FTF configuration file’s FileTypes property allows you to enable processing for data that does not reside in files.

To work with data that does not exist in files, you must specify the following property:

- **FileTypes** – The connector invoked for each valid file type.

Unlike other property values, which take a single value or comma-delimited set of values, the FileTypes property uses a more complex syntax. The following model displays the FileTypes property’s syntax:

```
FileTypes = sdrFileTypeSpec SDR sdrPortalDLL sdrPortalEntryPoint,  
           rcvFileTypeSpec RCVR rcvPortalDLL rcvPortalEntryPoint
```

Where:

- *sdrFileTypeSpec* – The file type value to be used in the calling interface to call the specified Sender connector. **Recommended value:** AS400PF
- *sdrPortalDLL* – The Sender connector DLL invoked to work with the specified data type.

## Working with AS/400 Files

### Using the FTF Command

- *sdrPortalEntryPoint* – The entry point for the Sender connector DLL.
- *rcvFileTypeSpec* – The file type value to be used in the calling interface to call the specified receiver connector. **Recommended value:** AS400PF
- *rcvPortalDLL* – The Receiver connector DLL invoked to work with the specified data type.
- *rcvPortalEntryPoint* is the entry point for the Receiver connector DLL.

### Example

In the following example, Sender and Receiver processing is specified for AS/400 physical files.

```
FileTypes = AS400PF SDR PTLAS400 PTLAS400SEND,  
            AS400PF RCVR PTLAS400 PTLAS400RECEIVE
```

## Using the FTF Command

To use the command-line interface to send or receive AS/400 physical or logical file data, you must use the FTF command. This section describes the FTF command and its parameters that relate specifically to AS/400 logical and physical files.

This section does not include all the FTF command parameters used to run a transfer request. For complete FTF command syntax, see the *Tivoli Data Exchange Technical Reference*, “Interface Commands.”

## **Sending AS/400 Physical or Logical File Data**

To send AS/400 physical or logical file data, use the FTF command with the following parameters:

- **-stype** *sourceType* – The type of the source data being sent in the transfer. **Valid values:** any data type specified in the configuration file. This value is case sensitive. **Recommended value:** AS400PF
- **-dtype** *destType* – The type of destination data being created as the result of the transfer. **Valid values:** any data type specified in the configuration file. This value is case sensitive. **Recommended value:** AS400PF
- **-sdata** *sourceData* – Source of the data being sent in the transfer.
- **-ddata** *destData* – Destination of the data being created as the result of the transfer.

---

### **Notes:**

- You can use this ability to send data not stored in files, receive it, or both send and receive it. In other words, you can use FTF to convert data from one format to another during the transfer.
  - When you submit a data-transfer request involving AS/400 physical or logical files, capture the transfer's FTFID (displayed when you execute the data-transfer request) to use in retrieving the transfer's status. You must use the FTFSTAT function to get information about the success or failure of a transfer involving AS/400 physical or logical files.
- 

### **sdata and ddata Parameters**

For transfers involving AS/400 physical and logical files, the FTF sdata and ddata parameters require the following syntax:

```
`TBL=tableName;RCDFMT=recordFormat;FIXLEN=isFixed;  
DLM=delimiterChar;'
```

## Working with AS/400 Files

### Using the FTF Command

Where:

- **TBL**=*tableName* contains the name of the source or target table. This value should be specified in the following format: library/file(member). You should specify this parameter for each AS/400 physical or logical file being sent or received.
- **RCD\_FMT**=*recordFormat* is the data's source or target record format. If you do not specify a record format, the first record format is used.
- **FIXLEN**=*isFixed* determines whether the source or target file information is fixed length (Y) or variable (N). You should specify this value only when the file on the other end of the transfer is a text file.  
**Default:** Y
- **DLM**=*delimiterChar* determines the delimiter character for delimited text data. **Default:** , (comma)

In the following example, the source data for the data transfer is an AS/400 logical file. The records with the format of RFMT2 are being transferred to a delimited text file that uses an asterisk for the delimiter.

```
-sdata `TBL=PROD/WKLYSALE;RCD_FMT=RFMT2;FIXLEN=Y;  
DLM=*;`
```

## Examples

This section presents examples that describe how to use the FTF command to perform the following types of data-transfer requests:

- Sending physical or logical file data to a fixed-length text file (see page 112)
- Sending physical or logical file data to a delimited text file (see page 113)
- Sending fixed-length text data to a logical or physical file (see page 115)
- Sending delimited text data to a logical or physical file (see page 116)
- Sending physical file data to a logical file or another physical file (see page 117)
- Sending logical file data to a physical file or another logical file (see page 119)

Each example lists only the FTF command parameters required to understand the example. For complete syntax information about the FTF command, see “FTF” in the *Tivoli Data Exchange Technical Reference*.

## **Sending to a Fixed-Length Text File**

This section describes how to send AS/400 physical and logical files to a fixed-length text file. This ability not only allows you to transfer data from one location to another, it mirrors a database export function, creating a fixed-length text file at the specified destination.

The `-sdata FIXLEN=Y` parameter determines that the data will be written to the destination as a fixed-length text file.

When you run a data-transfer request with a fixed-length text file as its destination, FTF carries the field integrity from the source to the destination. In other words, if the source file includes a ten-character field, ten characters will be devoted to that field in the destination file, even if all ten characters are not filled. When FTF encounters data that does not fill a fixed-length data field, it uses blanks in the destination file to pad the field out to its specified length.

### **Sending from a Physical File**

In the following example, the FTF command transfers an AS/400 physical file to a fixed-length text file. When it reads the source file, FTF can tell that the file you are transferring is a physical file.

```
ftf -sqm PROD1 -dqm PROD6 -stype AS400PF -sdata  
'TBL=PROD/MONTOTAL;FIXLEN=Y' -dpath  
e:\temp\export.txt -type TEXT
```

### **Sending from a Logical File**

In the following example, the FTF command transfers an AS/400 logical file to a fixed-length text file. When it reads the source file, FTF can tell that the file you are transferring is a logical file. Because the `-sdata RCDFMT` parameter is included, FTF processes only those records with the specified record format. If no record format is specified, FTF processes the records with the first record format.

```
ftf -sqm PROD1 -dqm PROD6 -stype AS400PF  
-sdata 'TBL=PROD/MONTOTAL;FIXLEN=Y;RCDFMT=RFMT1'  
-dpath e:\temp\export.txt -type TEXT
```

## Results

The following figure contains an example of how this data would look if it were transferred to a fixed-length text file for which a file length of 10 was specified for a person's first name and last name. (Each record also contains a phone number and a pay rate [a floating-point value with two decimal places].)

RICHARD	OSHEA	518-555-1234+022.00
CONNIE	SISE	518-555-9763+101.00
JUSTIN	KASITCH	703-555-9806+031.55

## Sending to a Delimited Text File

This section describes how to send AS/400 physical and logical files to a delimited text file. FTF creates a delimited text file at the destination under the following conditions:

- The FTF command's -sdata parameter contains a FIXLEN value of N.
- The FTF command's -sdata parameter does not contain a FIXLEN value (delimited is the default for transfers to a text file).

If you do not use the -sdata DLM value to specify a delimiter, FTF uses a comma as the delimiter character.

## Sending from a Physical File

In the following example, the FTF command transfers an AS/400 physical file to a fixed-length text file. In this case, because the delimiter is a comma, no delimiter value is required.

## Working with AS/400 Files

### Examples

```
ftf -sqm PROD1 -dqm PROD6 -stype AS400PF  
-sdata 'TBL=PROD/MONTOTAL;'  
-dpath e:\temp\export.txt -type TEXT
```

### Sending from a Logical File

In the following example, the FTF command transfers an AS/400 logical file to a delimited text file. In this case, the delimiter is specified as an asterisk.

```
ftf -sqm PROD1 -dqm PROD6 -stype AS400PF  
-sdata 'TBL=PROD/MONTOTAL;RCDFMT=RFMT1;DLM=*'  
-dpath e:\temp\export.txt -type TEXT
```

### Results

In the first example, the DLM parameter is an asterisk. This asterisk determines the end of each field when the destination file is written. The following figure contains an example of what data would look like if it were transferred from a physical file to an asterisk-delimited text file.

```
"RICHARD"*"OSHEA"*"518-555-1234"*+22.00  
"CONNIE"*"SISE"*"518-555-9763"*+101.00  
"JUSTIN"*"KASITCH"*"703-555-9806"*+31.55
```

In the second example, the DLM parameter is omitted, resulting in a comma being used as the delimiter in the destination file. The following figure contains the same data in a comma-delimited text file.

```
RICHARD,OSHEA,518-555-1234,+22.00  
CONNIE,SISE,518-555-9763,+101.00  
JUSTIN,KASITCH,703-555-9806,+31.55
```



## **Sending from a Fixed-Length Text File**

FTF allows you to send the contents of fixed-length text files to AS/400 physical or logical files. This ability not only allows you to transfer data from one location to another, it mirrors a database import function.

---

### **Note:**

Because the data in the flat file has no accompanying information about its structure and data types, you must create the target physical or logical file before you send data to it from a text file.

---

When FTF writes to a file, it can determine whether the file to which it is writing is a physical file or a logical file.

When you send data from a fixed-length file to an AS/400 physical or logical file, you must set the `-ddata FIXLEN` value to `Y`, indicating that the data being sent is fixed-length. You should also consider verifying that the lengths that exist in the source file match the lengths specified for the target file.

### **Sending to a Physical File**

In the following example, the FTF command is used to transfer data from a fixed-length text file to a physical file.

```
ftf -sqm PROD1 -dqm PROD6 -spath e:\temp\export.txt  
-type TEXT -dtype AS400PF  
-ddata 'TBL=PROD/MONTOTAL;FIXLEN=Y;'
```

### **Sending to a Logical File**

In the following example, the FTF command is used to transfer data from a fixed-length text file to a logical file. When you transfer data to a logical file, FTF can only write to the first record format in that file. As a result, the `-ddata RCD_FMT` parameter is unnecessary. You should consider using target files with only one record format.

```
ftf -sqm PROD1 -dqm PROD6 -spath e:\temp\export.txt  
-type TEXT -dtype AS400PF  
-ddata 'TBL=PROD/MONTOTAL;FIXLEN=Y;'
```

## **Sending from a Delimited Text File**

FTF allows you to send the contents of delimited text files to AS/400 physical or logical files. This ability not only allows you to transfer data from one location to another, it mirrors a database import function.

---

### **Note:**

Because the data in the flat file has no accompanying information about its structure and data types, you must create the target physical or logical file before you send data to it from a text file.

---

When you send data from a delimited file to an AS/400 physical or logical file, you can use the -ddata DLM parameter to specify a delimiter of your choosing, or you can omit the DLM parameter and use a comma as a delimiter (the FTF default). When the specified delimiter character is encountered, FTF writes the data to the next column in the table.

---

### **Note:**

When you specify a delimiter for your file, make sure the delimiter character you select is not a naturally occurring character in that file. If the delimiter character exists as a natural part of the data, it will cause the data to be inserted in the column incorrectly.

---

## **Sending to a Physical File**

In the following example, the FTF command transfers data from an asterisk-delimited text file to a physical file.

```
ftf -sqm PROD1 -dqm PROD6 -spath e:\temp\export.txt  
-type TEXT -dtype AS400PF  
-ddata 'TBL=PROD/MONTOTAL;DLM=*;'
```

## **Sending to a Logical File**

In the following example, the FTF command transfers data from a comma-delimited file to a logical file. Because the comma is the default delimiter character, the -ddata DLM parameter is omitted. When you write to a logical file, FTF can only write to the first record format in that file. As a result, the -ddata RCDfmt parameter is unnecessary. You should consider using target files with only one record format.

```
ftf -sqm PROD1 -dqm PROD6 -spath e:\temp\export.txt  
-type TEXT -dtype AS400PF  
-ddata 'TBL=PROD/MONTOTAL;'
```

## **Sending AS/400 Physical Files**

FTF allows you to send AS/400 physical files. Information about sending physical files to text file was covered in previous sections. This section describes the process for sending an AS/400 physical file to another AS/400 physical file or to an AS/400 logical file.

### **Creating the Destination Physical File**

When you are sending an AS/400 physical file to a physical file, the destination file does not have to exist. If the destination file does not exist, FTF can use information sent with the data to create it and specify its structure and data types. You can also use FTF to send AS/400 physical and logical files to files that already exist.

**Note:**

Data type matching in FTF is flexible. When you transfer AS/400 physical or logical data to an existing file, FTF does not require the data types to match exactly. For instance, if you send varchar to a table in which the corresponding data is of type char, FTF does not prevent the transfer from occurring.

**Sending to a Physical File**

In the following example, the FTF command sends a physical file to another physical file. In this example, the target file does not exist, so FTF creates it, based on the information sent as part of the transfer.

```
FTF -sqm PROD1 -dqm PROD9 -stype AS400PF
-sdata 'TBL=TEST/MONDATA' -dtype AS400PF
-ddata 'TBL=PROD/MONDATA'
```

The FTF data conversion algorithm allows flexibility in data types during transfers. The following table lists the data conversions that are made during the data transfer.

Source Type	Destination Type
ZONED	FLOAT
PACKED	FLOAT

This data conversion allows you to use the destination data in relational databases, such as Oracle and Sybase, at a future time.

**Sending to a Logical File**

In the following example, the FTF command sends a physical file to a logical file. In this example, the target file has the same data types as the source file.

---

### **Notes:**

- When you transfer data to a logical file, FTF can only write to the first record format in that file. You should consider using target files with only one record format. If the data you are sending does not match the target record format, errors can occur.
  - When you transfer data to a logical file, the logical file must exist before the data transfer occurs.
- 

```
FTF -sqm PROD1 -dqm PROD9 -stype AS400PF  
-sdata 'TBL=TEST/MONDATA' -dtype AS400PF  
-ddata 'TBL=PROD/LFMDATA'
```

## **Sending AS/400 Logical Files**

FTF allows you to send AS/400 logical files. Information about sending logical files to text file was covered in previous sections. This section describes the process for sending an AS/400 logical file to another AS/400 logical file or to an AS/400 physical file.

### **Creating the Destination Physical File**

When you are sending an AS/400 logical file to a physical file, the destination file does not have to exist. If the destination file does not exist, FTF can use information sent with the data to create it and specify its structure and data types. You can also use FTF to send AS/400 physical and logical files to files that already exist.

---

**Note:**

Data type matching in FTF is flexible. When you transfer AS/400 physical or logical data to an existing file, FTF does not require the data types to match exactly. For instance, if you send varchar to a table in which the corresponding data is of type char, FTF does not prevent the transfer from occurring.

---

### **Sending to a Physical File**

In the following example, the FTF command is used to send a logical file to a physical file. In this example, the target file does not exist, so FTF creates it based on the information sent as part of the transfer.

```
FTF -sqm PROD1 -dqm PROD9 -stype AS400PF  
-sdata 'TBL=TEST/MONDATA' -dtype AS400PF  
-ddata 'TBL=PROD/MONDATA'
```

The FTF data conversion algorithm allows flexibility in data types during transfers. The following table lists the data conversions that are made during the data transfer.

Source Type	Destination Type
ZONED	FLOAT
PACKED	FLOAT

This data conversion allows you to use the destination data in relational databases, such as Oracle and Sybase, at a future time.

### **Sending to a Logical File**

In the following example, the FTF command is used to send a logical file to another logical file. In this example, the target file has the same data types as the source file.

---

### Notes:

- When you transfer data to a logical file, FTF can only write to the first record format in that file. You should consider using target files with only one record format. If the data you are sending does not match the target record format, errors can occur.
  - When you transfer data to a logical file, the logical file must exist before the data transfer occurs.
- 

```
FTF -sqm PROD1 -dqm PROD9 -stype AS400PF  
-sdata 'TBL=TEST/MONDATA' -dtype AS400PF  
-ddata 'TBL=PROD/LFMDATA'
```

## Determining the Transfer's Success

In FTF transfers that involve AS/400 physical and logical files, you must use the FTFSTAT command to determine the status of the transfers, rather than the command-line responses from FTF. This section describes the FTFSTAT command used to get status information on an AS/400 system and describes the output that indicates an error in executing the transfer.

To get transfer status on an AS/400 system, use the following command:

```
FTFSTAT LQM(localQueueMgr) CFILE(configFile)  
FTFID(ftfIDVal) FORMAT(*DETAIL)
```

Where:

- *localQueueMgr* – The lqm value for which status records are returned.
- *configFile* – The fully qualified path and filename for the FTF configuration file. On OS/390 platforms, if no cq argument is specified, this value must be specified. In other operating systems, it is optional if the FTF\_CONFIG\_FILE environment variable is set.

## Working with AS/400 Files

### *Determining the Transfer's Success*

- *ftfIdValue* – The FTFID value for which status records are returned. The FTFSTAT command normally displays all status records unless this argument is used. To get a partial list displayed, wildcards can be used. An asterisk can be placed in the character strings at any point to get the records that match the characters entered until the asterisk is encountered.

For complete information about the FTFSTAT command, see “FTFSTAT” in the *Tivoli Data Exchange Technical Reference*.

If the transaction completed successfully, the status information should indicate that fact. The following sections list the FTFSTAT information that appears if the data transfer failed when AS/400 physical or logical file data was sent and when AS/400 physical or logical file data was received.

## AS/400 Sender Status Errors

If the transfer failed when AS/400 physical or logical file data was sent, the following information appears in the FTFSTAT output:

```
Timestamp           : timeStamp
QMGr                 : queueMgrName
Component            : AS/400 Sender Connector
Status               : ERROR_PROCESSING
Error                : 0
errorMsg
```

Where:

- *timeStamp* – The date and time when the error occurred.
- *queueMgrName* – Name of the queue manager where the error occurred.
- *errorMsg* – The error message associated with the failure. The following error messages can be returned as the result of a failure sending AS/400 physical or logical file data:
  - Invalid source file – The source file specified in the data-transfer request is not a valid AS/400 physical or logical file.
  - Unable to validate source file – FTF could not determine whether the source file was a physical file, a logical file, or another type of file. This error condition can result from a system API error.



- Source file is not a physical or logical file – The source file was of a type other than physical or logical.
- Failed to retrieve table column information – FTF could not retrieve information concerning the source table's columns. This error condition can result from a system API error.
- Failed to retrieve DTYPE information – FTF could not retrieve the information specified in the FTF command's dtype parameter. This information is required to determine whether the target file requires information about the structure of the source file.
- Failed to send table schema – FTF could not send information about the structure of the table data being sent. This error condition can result from communications errors.
- Failed to retrieve table data – FTF could not retrieve the table data being sent. As a result it could not send the data to the destination. This error condition can result from an error attempting to read the source file, for example when it is locked by another user.

## Example

In the following example, FTF could not send the source data because the source file was not an AS/400 logical or physical file.

Timestamp	: 12/21/00 12:11:30
QMgr	: PROD1
Component	: AS/400 Sender Connector
Status	: ERROR_PROCESSING
Error	: 0
Source file is not a physical or logical file	

## AS/400 Receiver Status Errors

If the transferred failed when AS/400 physical or logical file data was received, the following information appears in the FTFSTAT output:

## Working with AS/400 Files

### *Determining the Transfer's Success*

```
Timestamp           : timeStamp
QMgr                : queueMgrName
Component           : AS/400 Receiver Connector
Status              : ERROR_PROCESSING
Error               : 0
errorMsg
```

Where:

- *timeStamp* – The date and time at which the error occurred.
- *queueMgrName* – Name of the queue manager on which the error occurred.
- *errorMsg* – The error message associated with the failure. The following error messages can be returned as the result of a failure receiving AS/400 physical or logical file data:
  - Invalid member name. Use ADDPFM or ADDLFM to add a member – The member name specified in the data-transfer request does not exist.
  - Unable to validate target file – FTF could not determine whether the target file was a physical file, a logical file, or another type of file. This error condition can result from a system API error.
  - Target file is not a physical or logical file – The target file was of a type other than physical or logical.
  - Failed to retrieve STYPE information – FTF could not retrieve the information specified in the FTF command's stype parameter. This information is required to determine whether information is required about the structure of the source file.
  - Failed to receive source table schema – FTF did not receive information about the structure of the table data being sent. This error can result from a communications error.
  - Cannot create the target table without source table schema – The information required to create the physical or logical data was not included with the data being transferred. This error can occur when the source is not of a format that includes schema, such as a text file format. This error can occur if you try to send a text file to a physical file that does not exist.

- Failed to create target table based on the source table schema – The schema sent for the source data was not sufficient to create the target table.
- Failed to retrieve target table schema – FTF could not retrieve the target table's schema. This error can result from a read error.
- Target file is not compatible with the source file – The structure or data types of the target file and the source file are not compatible.
- Failed to load data into target table – The FTF Receiver could receive the data, but could not load it into its target table.

## Example

In the following example, FTF received the data but could not load it into a target table.

Timestamp	: 12/21/00 12:11:30
QMgr	: PROD1
Component	: AS/400 Receiver Connector
Status	: ERROR_PROCESSING
Error	: 0
Failed to load data into its target table	



# Using the ISPF User Interface

Tivoli Data Exchange (TDE) includes ISPF panels that allow you to interactively submit and monitor data transfers in an OS/390 environment. These panels allow you to interactively perform all the tasks the TDE (FTF) commands and APIs perform.

The ISPF panels are useful for FTF administrators monitoring data-transfer requests and submitting ad hoc requests. They are also useful for developers who want to work out the details of a particular data-transfer request before writing code or scripts to perform the transfer.

This chapter describes how to access and use the FTF ISPF panels. It includes the following information:

<b>Section</b>	<b>Page</b>
ISPF Panel Structure	128
Accessing the ISPF Panels	129
Requesting a Data Transfer	129
Staging Data-Transfer Requests	153
Viewing Status Information	155
Cancelling a Data-Transfer Request	160
Pinging Components	161
Shutting Down Components	164
Exiting the ISPF Panels	165

Assumptions

This section makes the following assumptions:

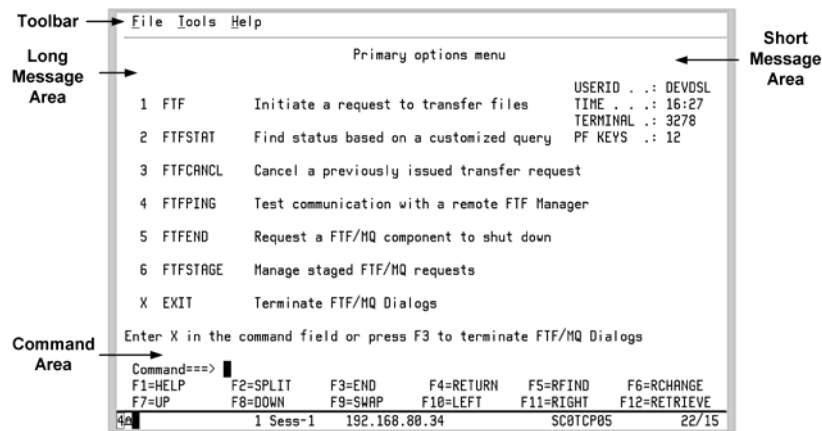
- The ISPF panels have been successfully installed on an IBM mainframe computer that has access to the file being transferred and to all FTF components. For information about installing the ISPF panels, see "Initializing the ISPF User Interface" in the *Tivoli Data Exchange Installation Guide*.
- You have appropriate administrative privileges for the functionality in the ISPF panels.

ISPF Panel Structure

The ISPF panels that compose the user interface have a consistent structure. This section describes these parts of the structure:

- Toolbar – Contains dropdown menus to execute FTF functions and to exit the user interface.
- Command area – Allows you to enter command options.
- Short message area – Displays terse status information.
- Long message area – Displays more detailed status information

The following diagram displays and labels each listed area.



You can use the toolbar to activate the FTF commands listed on the main FTF panel, get help, and exit the FTF ISPF user interface. The exit function allows you to exit from any command panel in the user interface.

You can use the command area to enter panel commands. For instance, in the panel displayed above, you can enter the number corresponding to a primary option and press **[ENTER]** to execute that option.

When a FTF option is executed, a results message appears in the short message area. Press **[PF1]** to display a more detailed message in the long message area. When the long message appears, you can press **[PF1]** to display help information about the command associated with the status information.

## Accessing the ISPF Panels

If the ISPF panels do not appear in your primary options menu, contact your FTF administrator for information about how to invoke them.

## Requesting a Data Transfer

The FTF option on the *Primary Options Menu* panel allows you to request a data transfer. When you use the ISPF panels to request a data transfer, you are performing the same task as if you use the FTF command or the FTFReq function in the FTF API set.

The screenshot shows the 'Primary options menu' panel. At the top is a menu bar with 'File', 'Tools', and 'Help'. Below it, the title 'Primary options menu' is centered. The main area contains a list of options, each with a number, a command name, and a description. To the right of this list, user information is displayed: USERID . . : DEVDSL, TIME . . . : 16:27, TERMINAL . : 3278, and PF KEYS . : 12. Below the list, a message states: 'Enter X in the command field or press F3 to terminate FTF/MQ Dialogs'. At the bottom, there is a 'Command====>' field followed by a cursor. Below this field is a row of function key definitions: F1=HELP, F2=SPLIT, F3=END, F4=RETURN, F5=RFIND, F6=RCHANGE, F7=UP, F8=DOWN, F9=SWAP, F10=LEFT, F11=RIGHT, and F12=RETRIEVE. At the very bottom, a status bar displays: '1 Sess-1', '192.168.80.34', 'SC0TCP05', and '22/15'.

Option	Command	Description
1	FTF	Initiate a request to transfer files
2	FTFSTAT	Find status based on a customized query
3	FTFCANCL	Cancel a previously issued transfer request
4	FTFPING	Test communication with a remote FTF Manager
5	FTFEND	Request a FTF/MQ component to shut down
6	FTFSTAGE	Manage staged FTF/MQ requests
X	EXIT	Terminate FTF/MQ Dialogs

Enter X in the command field or press F3 to terminate FTF/MQ Dialogs

Command====> █

F1=HELP F2=SPLIT F3=END F4=RETURN F5=RFIND F6=RCHANGE  
F7=UP F8=DOWN F9=SWAP F10=LEFT F11=RIGHT F12=RETRIEVE

1 Sess-1 192.168.80.34 SC0TCP05 22/15

## Using the ISPF User Interface

### Requesting a Data Transfer

To request a data transfer, follow these steps:

1. In the *Primary options menu* panel, type **1** or select **Tools>Initiate Request**. The *Initiate a request to transfer files* panel appears. The FTFID value is automatically generated when the data-transfer request is submitted.

```
File  Tools  Help
-----
Initiate a request to transfer files

COMMAND ===>

FTF Id:                                     User: SQADRC
Originating QMGR . . . . . TECH1
Source QMGR . . . . . TECH1
Destination QMGR . . . . . TECH1
Label . . . . .
Source path . . . . . C:\TEXT

Destination Path . . . C:\TEXT

Specify additional options.
Enter X next to the following for additional options.
- Transfer options                      - MVS/ESA Allocation options
- Application Integration Options        - AS/400 File options
- Stage options                         - Connector Software options
- Reply/Notify options
```

2. Enter the following information, as appropriate:
  - **Originating QMGR** – Queue manager on which the data transfer's FTF Manager operates. You must enter an Originating QMGR value.
  - **Source QMGR** – Queue manager on which the data transfer's FTF Sender operates. You must enter a Source QMGR value.
  - **Destination QMGR** – Queue manager on which the data transfer's FTF Receiver operates. You must enter a Destination QMGR value.
  - **Label** – The data-transfer request's label. Labels are user-specified qualifiers you can use to create data-transfer groupings for viewing file status information.
  - **Source path** – The fully qualified path and filename of the source file. You must enter a source path value.
  - **Destination Path** – The fully qualified path and filename of the target file. You must enter a destination path value.



3. Type an **X** in front of the appropriate options for entering additional entry options. None of these options is required. If you select more than one set of options, you are taken from one set of options to the next until all options are set. You can initiate the data-transfer request from any of the options panels.

From this panel, you can access panels for the following additional data entry:

- **Transfer options** (see “Entering Transfer Options” on page 132)
  - **Application Integration Options** (see “Entering Application Integration Options” on page 138)
  - **Stage options** (see “Setting Stage Options” on page 141)
  - **Reply/Notify options** (see “Setting Reply/Notify Options” on page 144)
  - **MVS/ESA Allocation options** (see “Setting MVS/ESA (OS/390) Allocation Options” on page 145)
  - **AS/400 File options** (see “Setting AS/400 File Options” on page 148)
  - **Connector Software options** (see “Setting Connector Software Options” on page 149)
4. Perform one of the following steps:
    - Press **[ENTER]** to submit the data-transfer request or move to the first set of options.
    - Press **[END]** to cancel the entries and clear the panel.
    - Press **[PF3]** to return to the *Primary Options* panel.

**Note:**

When a data-transfer request fails and the file mode for the FTF Receiver is set to “Create” or “Replace,” the Receiver component deletes the file. If the transfer request fails before the Receiver starts processing to the target file, cleanup is not required. However, if the mode is set to “Create” or “Replace” and the Receiver has started writing data to the target file when the transfer request fails, the Receiver deletes the file.

**Entering Transfer Options**

Transfer options govern how the data-transfer request is processed. You can set transfer options from the *Data Transfer and Performance Options* panel.

FileToolsHelp

Data Transfer and Performance Options

Panel: FTF10002  
User : SQADRC

COMMAND ==>

Mode . . . 1 1. Create  
2. Append  
3. Noreplace

Wrap . 1 1. Wrap  
2. Truncate  
3. Fail

Type . . 1 1. Binary  
2. Text

Pad . . . \_ 1. Pad  
2. No Pad

Pad Char . \_\_\_\_\_

Cancel Mode . 1 1. Neutral  
2. On  
3. Off

Priority . . . . . 5 (1-5)

Pool Name . . . . . \_\_\_\_\_

Expiry Timeout . . . . . \_\_\_\_\_ (in seconds)

Message Size . . . . . \_\_\_\_\_ (times KB)

Stage Type . . . . . 1 1. Non-persistent  
2. Persistent

Enter "/" to select option

— Make Directory

— Compress Data

— Trusted Transfer

— Persistent Data messages

— Stage Data Messages

— Delete Source

— Immediate

— DBCS Processing

Press ENTER to continue or END to cancel changes.

## Accessing the Transfer Options Panel

To access the *Data Transfer and Performance Options* panel, follow these steps:

1. At the *Primary options menu*, choose **FTF**.
2. At the *Initiate a request to transfer files* panel, enter the Originating QMGR and the Source QMGR.
3. Place an **X** next to the Transfer options selection and press **[ENTER]**. The *Transfer Options* panel appears.

To specify transfer option values, follow these steps:

1. Enter the following values as appropriate:
  - **Mode** – Determines what happens to the file when it reaches its destination. The following table lists and describes the transfer mode options. **Default value:** 1 (create).

Option	Name	Description
1	Create	Creates the target file if it does not already exist. If the target file does exist, overwrites it.
2	Append	Creates the target file if it does not already exist. If the target file does exist, appends the transferred data.
3	Noreplace	Creates the target file if it does not already exist. If the target file does exist, does not overwrite it and the data transfer fails.

- **Wrap** – The record processing options if the target record length is shorter than the source file record length. The following table lists and describes the wrap options.

Option	Name	Description
1	Wrap	Data that does not fit into the smaller record is written into the next record.
2	Truncate	Extra data is truncated and written to the target file.
3	Fail	The data transfer terminates with an error message.

## Using the ISPF User Interface

### *Requesting a Data Transfer*

- **Type** – The type of file being transferred. The following table lists and describes the file type. **Default value:** 1

Option	Name	Description
1	Binary	The file being transferred is binary.
2	Text	The file being transferred is text.

- **Pad** – Indicates if the record is padded or not padded. The following table lists and describes the record pad options.

Option	Name	Description
1	Pad	Specifies that blanks are to be inserted in each record to fill it out to the length of the other records in the file.
2	Nopad	Specifies that blanks are not to be inserted in each record to fill it out to the length of the other records in the file.

## **How Padding Works**

FTF padding functionality is governed at the source node (the FTF Sender) and by the characteristics of the data being transferred. No padding decisions are made at the target node (the FTF Receiver). If you request a data transfer in which the source and target nodes both run under operating systems that support record-formatted I/O (such as OS/390), using padding may cause undesired results if you specify different record lengths for your source and target files.

For example, if you send an OS/390 source file with a record length of 80 bytes to an OS/390 target file with a record length of 100 bytes, and if you enable padding, the target file's records are padded up to 80 bytes and contain blanks thereafter.

In addition, if you use the wrap option and the padding option, the padded characters wrap if the record length of the target file is less than the source file's record length. You can specify padding in the FTF command, the FTF configuration file, and the ISPF panels.

**Pad Character** – Determines the characters added to each record when data is transferred from an operating system that supports record-structured files to an operating system that does not. Applies only to text mode transfers. **Valid values:** an alphanumeric character, 0x00-0xFF (where the last two positions are a hexadecimal value mapped to the desired character), or a space. **Default value:** Space

---

### **Note:**

The hexadecimal character specified as a padding character is subject to MQSeries data translation rules. The hexadecimal character is converted according to the specification of the target system's code page.

---

## Using the ISPF User Interface

### *Requesting a Data Transfer*

- **Cancel Mode** – Determines whether the transaction can be preemptively cancelled. The following table describes the cancel mode options. **Default value:** Neutral.

Option	Name	Description
1	Neutral	Uses the value specified in the configuration file.
2	ON	Transaction can be preemptively cancelled.
3	OFF	Transaction cannot be preemptively cancelled.

### Preemptive Cancellation

Preemptive cancellation terminates a data-transfer request while it is still in progress. This type of cancellation adds overhead to the data-transfer process, but saves you from having to wait until the transfer is complete to cancel it. Preemptive cancellation is useful when you are transferring large amounts of data and you think you might have to cancel the transfer.

- **Priority** – Priority applied to the data-transfer request. **Valid values:** 1 (highest) – 5 (lowest). **Default value:** 5
- **Pool Name** – Nn't specify a pool, the default pool established in the FTF configuration file is used.
- **Expiry Timeout** – Time interval in minutes after which the data-transfer request is marked as expired if it is still running. If the transfer request times out, the FTFRCI\_REQUEST\_EXPIRED error code is returned.
- **Message Size** – Sets a message size value that overrides the MQSeries message-size value. This value is set in KB. **Valid values:** 1-3906 KB
- **Stage Type** – Determines whether a message is persistent. The following table lists and describes the stage type options.

<b>Option</b>	<b>Name</b>	<b>Description</b>
<b>1</b>	Non-persistent	Specifies that the messages in the staging queue are not persistent. If you specify this argument, the messages do not exist after a system or FTF reboot or shutdown.
<b>2</b>	Persistent	Specifies that the messages in the staging queue are persistent. If you specify this argument, the messages still exist after a system or FTF reboot or shutdown. By selecting this argument, you increase the recovery ability of FTF but reduce performance.

2. This panel allows you to set additional options by placing a slash in front of them. The following table lists and describes the options you can set in this manner.

<b>Option</b>	<b>Description</b>
Make Directory	If you select this option and the specified target directory structure does not exist, FTF creates it.
Compress Data	If you select this option, the data being sent is compressed using the FTF compression algorithm.
Trusted Transfer	If you select this option, FTF sacrifices data-recovery ability to allow for a faster transfer. This option should be used only when file recovery by means other than FTF is assured.
Persistent Data Messages	If you select this option, the messages being transferred are persistent. They still exist after a system or FTF shutdown or reboot.
Stage Data Messages	If you select this option, the data messages being transferred are stored in a staging queue until the data-transfer request is complete. Staging allows the file to be resent from the staging queue, rather than from the original file. Staging is appropriate if the data being transferred does not permanently exist on the source node.
Delete Source	Indicates that the source data is to be deleted once the data-transfer request is completed.

**Using the ISPF User Interface**  
*Requesting a Data Transfer*

Option	Description
Immediate	Gives you the ability to issue a FTF data transfer request that is processed synchronously between the Sender and Receiver rather than the normal asynchronous mode. Using this argument differentiates an immediate data transfer from a regular FTF transfer. Using FTF immediate, the FTF Receiver begins to process the data transfer request as soon as it gets the request instead of waiting for all of the data to arrive. Since the Receiver processes the data messages as they arrive, the queue storage required to transfer a large amount of data is significantly reduced. Normal FTF data transfers require that the Receiver have all related data messages before writing the data to the disk. For example, using FTF in normal mode processing very large data transfers such as 1 gigabyte, the receiving node needs to have 1 gigabyte for queue storage and 1 gigabyte to store the data. With FTF immediate, the storage space on the queue storage is not required.
DBCS Processing	Specifies the option to scan the source file and split each datablock at the point where double-byte validity is maintained. Using this option ensures that there is no truncation of a double-byte character when the specified message size does not span the entire character. If the DBCS option is used with the wrap option in a data-transfer request, the results can be unpredictable.

3. Perform one of the following steps:
- Press **[ENTER]** to submit the data-transfer request or move to the next set of options.
  - Press **[END]** to cancel the entries and clear the panel.
  - Press **[PF3]** to return to the *Initiate a request to transfer files* panel.

**Entering Application Integration Options**

User exits allow you to plug your own modules into FTF to execute custom business logic. For more information about creating user exits, see Chapter 5, “Tivoli Data Exchange User Exits.”



## Accessing the Application Integration Options Panel

To access the *Application Integration Options* panel, follow these steps:

1. At the *Primary options menu*, choose **FTF**.
2. At the *Initiate a request to transfer files* panel, enter the Originating QMGR and the Source QMGR.
3. Place an **X** next to the Application Integrations Options selection and press **[ENTER]**. The *Application Integration Options* panel appears.

The screenshot shows the 'Application Integration Options' panel. At the top, there is a menu bar with 'File', 'Tools', and 'Help'. Below the menu bar, the title 'Application Integration Options' is centered. On the left, it says 'COMMAND ===>'. On the right, it says 'User: SQADRC'. The panel contains two columns of input fields. Each column has four rows of labels: 'Exit Number . .', 'Exit DLL . . .', 'Exit Entry . .', and 'Exit Data . . .'. Each label is followed by a horizontal line for input. The first 'Exit Data' field in the left column has a small black square cursor. At the bottom left, it says 'Enter X for more exits'. At the bottom right, there is a label 'More Exits' with a small horizontal line to its left.

## Using the ISPF User Interface

### Requesting a Data Transfer

To specify application integration options values, follow these steps:

1. Enter the following values for each user exit you want to invoke:
  - **Exit Number** – Number of the exit you want to invoke. The following table lists the exits and their numbers. If you return to the *Initiate a Request to Transfer Files* panel, then return to this panel, you must reenter the exit number value.

FTF Exit No.	Executing Node	FTF Component	Description
Exit 3	originating queue manager (oqm)	FTF Manager	Manager preprocess exit. If specified, this is the first exit executed in the data-transfer request.
Exit 4	oqm	FTF Manager	Manager postprocess exit. If specified, this is the last exit executed in the data-transfer request.  <b>Note:</b> This exit's execution does not depend on the success or failure of the data-transfer request.
Exit 5	source queue manager (sqm)	FTF Sender	Sender preprocess exit. This exit is invoked before the FTF Sender reads the source file's contents.
Exit 6	sqm	FTF Sender	Sender postprocess exit. The FTF Sender invokes this exit after the source file's contents have been read and processed for transfer to the target node.  <b>Note:</b> This exit is invoked regardless of whether the FTF Sender is able to successfully process the source file.
Exit 7	destination queue manager (dqm)	FTF Receiver	Receiver preprocess exit. The FTF Receiver invokes this exit before it begins to write the target file.

FTF Exit No.	Executing Node	FTF Component	Description
Exit 8	dqm	FTF Receiver	Receiver postprocess exit. The FTF Receiver invokes this exit after the target file has been processed and all the source data has been written to it.  <b>Note:</b> This exit is invoked regardless of whether the target file is processed successfully.
Exit 9	sqm	FTF Sender	Invokes the Sender connector exit.
Exit 10	dqm	FTF Receiver	Invokes the Receiver connector exit. .

- **Exit DLL** – The DLL, shared object, load module, or service program that contains the exit module to be executed. This value cannot be longer than seven characters.
- **Exit Entry** – Name of the function in the DLL that contains the exit module.
- **Exit Data** – Parameter passed into the user exit module. If you use the sample exit that comes with FTF, this parameter contains the commands to be executed by the exit module.

If you need to specify more user exits, type an **X** in front of *More Exits* and press **[ENTER]**. Additional application integration panels appear, allowing you to enter more user exit information, until you have specified the maximum possible number of exits.

2. Perform one of the following steps:
  - Press **[ENTER]** to submit the data-transfer request.
  - Press **[END]** to cancel the entries and clear the panel.
  - Press **[PF3]** to return to the *Initiate a Request to Transfer Files* panel.

## Setting Stage Options

The stage option allows you to retrieve a source file from the staging queue or place data messages in a staging queue to await further processing.

## Accessing the Stage Options Panel

To access the *Stage Options* panel, follow these steps:

1. At the *Primary options menu*, choose **FTF**.
2. At the *Initiate a request to transfer files* panel, enter the Originating QMGR and the Source QMGR.
3. Place an **X** next to the Staged Files selection and press **[ENTER]**. The *Stage Options* panel appears.

```
File  Tools  Help

Stage Options

COMMAND ===>                                     User: SQADRC

Enter the Staged Request Information
Staged FTF Identifier . . . 
Staged Source file . . . 

Staging options
  _ Send from Stage          _ Stage Only
```

To specify stage options, follow these steps:

1. Enter the following values as appropriate:
  - **Staged FTF Identifier** – FTFID of the data-transfer request that will be sent from the staging queue. Use this option only if you are sending a file from the staging queue using the **Send from Stage** option.
  - **Staged Source File** – Name of the fully qualified path of the source file as it exists on the sending node.

- **Send from Stage** – Indicates that the source file specified in the **Staged Source File** field should be retrieved from the staging queue rather than from the actual source file.
- **Stage Only** – Places the data messages on the staging queue, but does not send them. If you specify this value, you cannot specify a destination file value.

## **Specifying the Staging Queue**

The staging queue is specified by the *SenderStageControlQueue* property in the FTF configuration file. The default value for this queue is FTFSDR.STAGE.CONTROL. To override this default, change the property value to the appropriate queue name and restart FTF.

---

### **Note:**

If you specify a filename for the transfer from the staging queue and multiple entries have the same filename, the file related to the first matching entry found is sent. If you need to specify a specific instance of a file in the staging queue, use that file's FTFID.

---

2. Perform one of the following steps:
  - Press **[ENTER]** to submit the stage request.
  - Press **[END]** to cancel the entries and clear the panel.
  - Press **[PF3]** to return to the *Initiate a request to transfer files* panel.

## Setting Reply/Notify Options

The reply/notify options allow you to specify the type of status message to be monitored and the disposition of the information. You may specify that replies be sent to a particular queue and queue manager. You may also specify information about actions to be taken if certain types of messages occur.

The screenshot shows a terminal window with a menu bar at the top containing 'File', 'Tools', and 'Help'. Below the menu bar is a title bar that reads 'Reply/Notify Options'. The main area of the window contains the following text:

```
COMMAND ===>                                     User: SQADRC

                                Reply Options

Reply Queue . . ■
Reply QMGR. . .

                                Notify Options

Notify Status . 
Notify Type . . 
Notify Data . . 

```

Enter the following values, as appropriate:

- **Reply Queue** – Name of the queue to which reply messages are to be routed.
- **Reply QMGR** – Name of the queue manager to which reply messages are to be routed.
- **Notify Status** – Defines when a notification message will be sent to the NotifyQueue (see *Tivoli Data Exchange Installation Guide*, "Tivoli Data Exchange Configuration: Notification Message Property" for more information). The notification is sent if the transaction's status matches the status specified in this argument. If you specify this argument, you must also specify *Notify Data* and *Notify Type* arguments. **Valid values:** Success, Failure, Nonsuccess (includes failed, cancelled, expired).

---

### Note:

FTF functions only deliver the notification messages; any further processing is the responsibility of the user.

---

- **Notify Type** – Specifies the user-defined method, such as EMAIL, PAGER, FAX, or WTO, that will be used to deliver a notification message based on a transaction's status. If you specify this argument, you must also specify *Notify Data* and *Notify Status* arguments.
- **Notify Data** – Specifies user-defined data to aid in notification, such as e-mail, pager, or fax information, that will be used to deliver a notification message based on a transaction's status. If you specify this argument, you must also specify *Notify Status* and *Notify Type* arguments.

## Setting MVS/ESA (OS/390) Allocation Options

MVS/ESA (OS/390) allocation options can be set for data transfers for which the target file will reside on an OS/390 system. Any OS/390 allocation option values set in the ISPF panels to override the OS/390 default values in the FTF configuration file.

### Accessing the MVS/ESA Allocation Options Panel

To access the *MVS/ESA Allocation Options* panel, follow these steps:

1. At the *Primary options menu*, choose **FTF**.
2. At the *Initiate a request to transfer files* panel, enter the Originating QMGR and the Source QMGR.
3. Place an **X** next to the MVS/ESA Allocation Options selection and press **[ENTER]**. The *MVS/ESA Target Dataset File Allocation Options* panel appears.

## Using the ISPF User Interface

### Requesting a Data Transfer

The screenshot shows a terminal window with a menu bar at the top containing 'File', 'Tools', and 'Help'. Below the menu bar, the title is 'MVS/ESA Target Dataset File Allocation Options'. The main area contains the following text:

```
COMMAND ===>                                     Panel: FTF10803
                                                    User : SQADRC

Destination Dataset Name:
  Organization . . . . . (PS, PDS )
  Directory Blocks . . . . .
  Record Format . . . . . (F, FB, V, VB)
  Record Length . . . . . (0 - 32767)
  Block Size . . . . . (0 - 32760)
  Allocation Unit . . . . . (BLK, TRK, CYL)
  Unit . . . . .
  Volser . . . . .
  Primary . . . . .
  Secondary . . . . .
  Model . . . . .
  Bufno . . . . . (0 - 255)

Press ENTER to continue or END to cancel changes.
```

### Specifying an Esoteric Unit Name

To specify an esoteric name for the OS/390 UNIT value, follow these steps:

- Do not set a value in the MVSVOLUME stanza in the FTF configuration file.
- Specify the *Unit* value in the *MVS/ESA Allocation Options* panel, or in the MVSUNITNAME stanza in the FTF configuration file on either the FTF Sender or the FTF Receiver.

To specify OS/390 data set allocation option values, follow these steps:

1. Enter the following values, as appropriate:
  - **Organization** – The file organization of the target file. Currently, physical sequential (PS) is the only valid organization value.
  - **Directory Blocks** – Sets up the number of directory blocks used in allocating the target PDS if it does not exist. If this argument is not specified, the value in the FTF configuration file is used. If neither is specified, the PDS allocation will fail. **Valid values:** 1-32760



- **Record Format** – The target file’s record format. **Valid values:** F (fixed), FB (fixed block), V (variable), VB (variable block)
  - **Record Length** – The target file’s logical record length. **Valid values:** 1 –32760
  - **Block Size** – The block size for the target file on OS/390. Specifying a block size of 0 enables the system to choose the optimum block size for the data set during allocation. If the record format is Fixed Block (FB), the block size in the blksize argument must be a multiple of the logical record length, the lrecl parameter. When the record format is variable block (VB), the blksize value must be at least four bytes greater than the lrecl value. **Valid values:** 0-32760
  - **Allocation Unit** – The target file’s allocation unit. **Valid values:** BLK (block), TRK (track), CYL (cylinder)
  - **Unit** – The unit name for the target file. This field’s value is installation-dependent. Obtain it from your OS/390 administrator.
  - **Volser** – The volume serial number for the target. This argument’s value is installation-dependent. Obtain it from your OS/390 administrator.
  - **Primary** – The number of primary allocation units required at the target.
  - **Secondary** – The number of secondary allocation units required at the target.
  - **Model** – A model data set for Generation Data Group (GDG) allocation. Consult your OS/390 administrator for the available model data sets.
  - **Bufno** – Allows you to specify the number of internal buffers that are to be used when processing data transfers. The throughput of an FTF data transfer is governed by a combination of the block size of the data being transferred and the number of buffers that are allocated for transfer in the bufno argument.
2. Perform one of the following steps:
- Press **[ENTER]** to submit the data-transfer request or move to the next set of options.
  - Press **[END]** to cancel the entries and clear the panel.

## Using the ISPF User Interface

### Requesting a Data Transfer

- Press [PF3] to return to the *Initiate a Request to Transfer Files* panel.

## Setting AS/400 File Options

If the target file will reside on an AS/400 system, you can set options to control the attributes of the target file.

### Accessing the AS/400 File Options Panel

To access the *AS/400 File Options* panel, follow these steps:

1. At the *Primary options menu*, choose **FTF**.
2. At the *Initiate a request to transfer files* panel, enter the Originating QMGR and the Source QMGR.
3. Place an **X** next to the AS/400 File Options selection and press [ENTER]. The *AS/400 File Creation Options* panel appears.

The screenshot shows a terminal window with a menu bar at the top containing 'File', 'Tools', and 'Help'. The title bar reads 'AS/400 File Creation Options'. Below the title bar, the text 'COMMAND ===>' is on the left and 'User: SQADRC' is on the right. The main content area contains the following text: 'Specify AS/400 File creation options for target file:', 'Destination File Name:', 'AS/400 File type . \_ 1. Source Physical', '2. Save File', 'Record Length . . \_\_\_\_ (13 - 32760) Enter "/" to select option', 'CCSID . . . . . \_\_\_\_ \_ Create Library', 'Library Text . . . . . \_\_\_\_', 'File Text . . . . . \_\_\_\_', 'Library Auxillary Storage Pool . . . \_\_\_\_', 'File Auxillary Storage Pool . . . . \_\_\_\_', and 'Press ENTER to cotinue or END to cancel changes.'

To specify AS/400 option values, follow these steps:

1. If you need to create a library as part of the data transfer, insert a slash (/) in front of the *Create Library* option.
2. Enter the following values, as appropriate:
  - **AS/400 File Type** – The target file’s AS/400 type. **Valid values:** 1 (Source Physical), 2 (Save file)
  - **Record Length** – Specifies the target file’s record length. **Valid values:** 13-32760
  - **CCSid** – Specifies the target file’s coded character-set identifier (CCSID).
  - **Library Text** – If a library is being created as part of this data transfer, determines the library’s description.
  - **File Text** – If the target file is being created as part of this data transfer, determines the file’s description.
  - **Library Auxiliary Storage Pool** – If a library is being created as part of this data transfer, determines its ASP.
  - **File Auxiliary Storage Pool** – If a file is being created as part of this data transfer, determines its ASP.
3. Perform one of the following steps:
  - Press **[ENTER]** to submit the data-transfer request or move to the next set of options.
  - Press **[END]** to cancel the entries and clear the panel.
  - Press **[PF3]** to return to the *Initiate a request to transfer files* panel.

## Setting Connector Software Options

Connectors are user exits created to extend FTF input and output options to data that is not contained in files. The *Connector Software Options* panel allows you to run connectors.

## Accessing the Connector Software Options Panel

To access the *Connector Software Options* panel, follow these steps:

1. At the *Primary options menu*, choose **FTF**.
2. At the *Initiate a request to transfer files* panel, enter the Originating QMGR and the Source QMGR.
3. Place an **X** next to the Connector Software Options selection and press **[ENTER]**. The *Connector Software Options* panel appears.

FileToolsHelp

Connector Software Options

COMMAND ===>

User: SQADRC

Source Options

Destination Options

Connector DLL .  
Entry Point . .  
Connector Data

Connector DLL .  
Entry Point . .  
Connector Data

OR

OR

Stype . . . . .  
Sdata . . . . .

Dtype . . . . .  
Ddata . . . . .

This panel allows you to specify source and destination connectors. Enter values in either the upper or lower portion of the panel, but not both portions of the panel.

To specify a connector, follow these steps:

1. Enter the following source options, as necessary:
  - **Connector DLL** – The DLL, shared object, load module, or service program that contains the exit module to be executed. This value cannot be longer than seven characters.
  - **Entry Point** – Name of the function of the DLL, shared object, load module, or service program that contains the connector module.
  - **Connector Data** – Parameter passed into the connector module.
2. Enter the following destination options, as necessary:
  - **Connector DLL** – The DLL, shared object, load module, or service program that contains the exit module to be executed. This value cannot be longer than seven characters.
  - **Entry Point** – Name of the function of the DLL, shared object, load module, or service program that contains the connector module.
  - **Connector Data** – Parameter passed into the connector module.
3. Enter the following source options, as necessary:
  - **Stype** – The type of source data to be transferred. This value must be the same as one of the FileTypes values specified in the FTF configuration file. **Recommended value:** AS400PF
  - **Sdata** – A string of values that determines the table name, the record format name, if the data is of a fixed length, and the delimiter character. **Sdata** takes the following form:  
  
“TBL=*tableName*;RCDFMT=*recordFormatName*;  
FIXLEN=*fixedLength*;DLM=*delimiterChar*”  
  
Where:
    - **TBL** *tableName* – Name of the physical or logical file located in *lib/filename(mbrname)*. **Default value:** If *mbrname* is not specified, the first member name is used.
    - **RCDFMT** *recordFormatName* – Source file’s record format. **Default value:** If this value is not specified, the first record format is used.

## Using the ISPF User Interface

### Requesting a Data Transfer

- **FIXLEN** *fixedLength* – Specifies whether the record will have a fixed length at the target file. **Valid values:** Y, N **Default value:** N
  - **DLM** *delimiterChar* – The character to be used as the delimiter. **Default value:** , (comma)
4. Enter the following destination options, as necessary:
- **Dtype** – The type of destination file into which the source file is to be transferred. This value must be the same as one of the FileTypes values specified in the FTF configuration file. **Recommended value:** AS400PF
  - **Ddata** – A string of values that determines the table name, the record format name, if the data is of a fixed length, and the delimiter character. **Ddata** takes the following form:

“TBL=*tableName*;RCDFMT=*recordFormatName*;  
FIXLEN=*fixedLength*;DLM=*delimiterChar*”

Where:

- **TBL** *tableName* – Contains the name of the physical or logical file located in lib/*filename(mbrname)*. **Default:** If *mbrname* is not specified, the first member name is used.
- **RCDFMT** *recordFormatName* – Determines the target file’s record format. **Default:** If this value is not specified, the first record format is used.
- **FIXLEN** *fixedLength* – Specifies whether the record will have a fixed length at the target file. **Valid values:** Y, N **Default:** N
- **DLM** *delimiterChar* – Contains the character to be used as the delimiter. **Default:** If this value is not specified, a comma is used.

5. Perform one of the following steps:
  - Press [ENTER] to submit the data-transfer request or move to the next set of options.
  - Press [END] to cancel the entries and clear the panel.
  - Press [PF3] to return to the *Initiate a Request to Transfer Files* panel.

## Staging Data-Transfer Requests

Staging the data-transfer request allows you to send the file from the staging queue rather than from the original file. Staging is appropriate if the data being transferred does not permanently exist on the source node. You can also use the *Manage Staged Requests* panel to query the staging queue for a list of its contents, purge all currently staged data-transfer requests, or purge specific staged data-transfer requests.

To access the *Manage Staged requests* panel, choose the FTFSTAGE option on the *Primary Options* menu. The *Manage Staged requests* panel appears.

File Tools Help					
<b>Manage Staged requests</b>					User: DEVDSL
Originating QMGR . . .					
Source QMGR . . . . .					
Action . . . . .	1. Query 2. Purge FTF Id 3. Purge Source File 4. Purge All				
FTF Id . . . . .					
Source path . . . . .					
Wait time . . . . .					
(in seconds)					
COMMAND ==>					
F1=HELP	F2=SPLIT	F3=END	F4=RETURN	F5=RFIND	F6=RCHANGE
F7=UP	F8=DOWN	F9=SWAP	F10=LEFT	F11=RIGHT	F12=RETRIEVE
4a	1 Sess-1	192.168.80.34	SC0TCP04	§ 6/26	

## Using the ISPF User Interface

### *Staging Data-Transfer Requests*

1. Enter the following information as appropriate:
  - **Originating QMGR** – Queue manager on which the data transfer's FTF Manager operates. You must enter an Originating QMGR value.
  - **Source QMGR** – Queue manager on which the data transfer's FTF Sender operates. You must enter a Source QMGR value.
  - **Query** – Determines that the staging queue will be queried. If you specify this argument, you cannot specify the Purge FTF Id, Purge Source File, Purge All, or FTF Id options.
  - **Purge FTF Id** – FTFID of the file that is purged from the staging queue. If you specify this argument, you must specify the FTF Id argument.
  - **Purge Source File** – Determines that the file specified in the FTFID option will be purged. If you specify this argument, you must specify the source file argument.
  - **Purge All** – Allows you to purge all files that are currently staged. If you use this option, all staged data-transfer requests are purged.
  - **FTF Id** – FTFID of the data-transfer request. If you specify this argument, you cannot specify the Query option.
  - **Source path** – Name of the fully qualified path of the source file as it exists on the sending node.
2. Specify the following:
  - **Wait time** – Determines the time in seconds to wait for a reply from the query operation. This value does not apply to the purge operation.  
**Default value:** 300 seconds (5 minutes)
3. Perform one of the following steps:
  - Press **[ENTER]** to submit the staging request.
  - Press **[END]** to cancel the entries and clear the panel.
  - Press **[PF3]** to return to the *Primary options menu*.



## Viewing Status Information

The FTFSTAT option on the *Primary options menu* panel allows you to view data-transfer request information for all or some of the current or past data-transfer operations for which status information is available.

To view status information, follow these steps:

1. From the *Primary options menu* panel, type **2** or select **Tools>Query Status**. The *Status Filter* panel appears. This panel allows you to enter filter information to reduce the status information returned.

```

      Status Filter
      FTF Identifier . . .
      Label . . . . .
      Date/Time Range (YYYYMMDDhhmmss)
      From . . . . .
      To . . . . .
      Queue Managers
      Originating . . .
      Source . . . . .
      Target . . . . .
      Source File . . . . .
      Target File . . . . .
      Status . . ☒ Active ☒ Complete ☒ Failed ☒ Cancelled ☒ Expired
      Command==>
      F1=HELP      F2=SPLIT    F3=END      F4=RETURN   F5=RFIND    F6=RCHANGE
      F7=UP        F8=DOWN    F9=SWAP     F10=LEFT   F11=RIGHT
  
```

4a 1 Sess-1 192.168.80.34 SC0TCP01 § 3/24

2. Enter the following filter information, as appropriate. Data-transfer status information must satisfy all filter criteria to appear as filtered status information.
  - **FTF Identifier** – FTFID for which status information is returned. You can use the wildcard value (\*) in specifying FTFIDs.
  - **Label** – The label for which status information is returned. Labels are user-defined identifiers created for grouping data-transfer requests for which you want to display status.

- **(Date/Time Range) From** – The beginning of the date range for which status information is returned. If you specify the beginning of the date range, but no end, FTF returns all status information logged since the specified date. **Format:** YYYYMMDDhhmmss, where YYYY is the four-digit year, MM is the two-digit month, DD is the day, hh is the hour (24-hour format), mm is the minute, and ss is the second.
- **(Date/Time Range) To** – The end of the date range for which status information is returned. If you specify the end of the date range, but no beginning, FTF returns all status information logged until the specified date. **Format:** YYYYMMDDhhmmss, where YYYY is the four-digit year, MM is the two-digit month, DD is the day, hh is the hour (24-hour format), mm is the minute, and ss is the second.

### Working with Date Values

The FTF ISPF panels allow you to specify a partial date value. To specify a partial date, you must specify everything to the left of the smallest specified value. In other words, if you want to specify an hour, you must also specify the day, the month, and the year.

- **(Queue Managers) Originating** – Originating queue manager for which status information is returned.
- **(Queue Managers) Source** – Source queue manager for which status information is returned.
- **(Queue Managers) Target** – Destination queue manager for which status information is returned.
- **Source File** – The fully qualified path and filename of the source file(s) for which status information is returned. You can expand the filter information by using wildcards in the source file designation.
- **Target File** – The fully qualified path and filename of the source file(s) for which status information is returned. You can expand the filter information by using wildcards in the target file designation.

3. Select the status types for which message information is included. To select a status, insert an **X** in front of the desired status type. To deselect a status, remove the **X** from in front of the desired status type. You can select any of the following status types:
  - Active
  - Complete
  - Failed
  - Cancelled
  - Expired
4. Perform one of the following steps:
  - Press **[ENTER]** to submit the status request.
  - Press **[END]** to cancel the entries and clear the panel.
  - Press **[PF3]** to return to the *Primary options menu*.

If you submitted the status request, the *Status List* panel appears, displaying the status information for any status records that satisfy the filter specifications

File Edit View Tools					
Status List				Row 1 to 2 of 2	
Command==> █				Scroll ==> PAGE	
Current View Option: Origination				USERID - DEVMCH	
Enter S for Summary, P for Purge or L for Logs in the command field				TIME - 16:01	
C	FTF/MQ Event	QMGr	Date	Time	Status
—	d3088681-9e96-11d2-a007-970dfe8e582b	P39B	981228	154938	REQUEST_COMPLE
—	0aea9381-9b50-11d2-9017-89401f76a141	P39B	981224	114524	REQUEST_COMPLE
***** Bottom of data *****					
<div> <div>F1=HELP</div> <div>F2=SPLIT</div> <div>F3=END</div> <div>F4=RETURN</div> <div>F5=RFIND</div> <div>F6=RCHANGE</div> <div>F7=UP</div> <div>F8=DOWN</div> <div>F9=SWAP</div> <div>F10=LEFT</div> <div>F11=RIGHT</div> <div>F12=RETRIEVE</div> </div>					
4a	1 Sess-1		192.168.80.34		SC0TCP01 4/15

## Using the ISPF User Interface

### Viewing Status Information

The following status information is displayed for each status record:

- **FTF/MQ Event** – FTFID for the status event logged in the current status record.
- **QMGr** – Name of the queue manager on which the event took place.
- **Date** – Date the current status information was logged. **Format:** YYMMDD, where YY is the two-digit year, MM is the month, and DD is the day.
- **Time** – Time at which the current status information was logged. **Format:** hhmmss, where hh is the hour (24-hour format), mm is the minute, and ss is the second.
- **Status** – Current record's status.

To navigate through status records, press **[PF7]** to scroll up and **[PF8]** to scroll down.

5. The following table lists and describes the actions available for processing status records. To perform one of the actions, type the appropriate letter in the *C* column (left-most column on the screen) and press **[ENTER]**.

Letter	Action
L	Displays log information about the data-transfer request of which the current status message was a part.
P	Purges the current record from the status information.
R	Resubmits the data-transfer request for which the status information appears.
S	Displays summary information about the data-transfer request for which the status information appears.

## Log Information Display

The log information panel displays log information for the data transfer that contains the current status record. If this information does not fit into the designated area, use **[PF7]** and **[PF8]** to scroll up and down (respectively) through the information. For each log entry, the log display contains the following information:

- Queue Manager – Queue manager responsible for the logged action.
- Timestamp – Date and time at which the logged action took place.
- Component – Component that performed the specified action.
- Status – Text and status number for the specified status condition.

## Summary Information Displays

The summary information panel displays summary information about the current status record. If this information does not fit into the designated area, use **[PF7]** and **[PF8]** to scroll up and down (respectively) through the information.

The summary display contains the following information:

- Timestamp – Date and time of the status information.
- Request QMgr – Name of the queue manager from which the request came.
- Originating QMgr – Name of the FTF Manager's queue manager.
- Status – Last status recorded at the FTF Manager.
- Status Time – Date and time the listed status was achieved at the FTF Manager.
- Source QMgr – Name of the FTF Sender's queue manager.
- Source File Name – The fully qualified path and name of the file on the FTF Sender.
- Status – Last status recorded at the FTF Sender.
- Status Time – Date and time the listed status was achieved at the FTF Sender.
- Target QMgr – Name of the FTF Receiver's queue manager.

**Using the ISPF User Interface**  
*Cancelling a Data-Transfer Request*

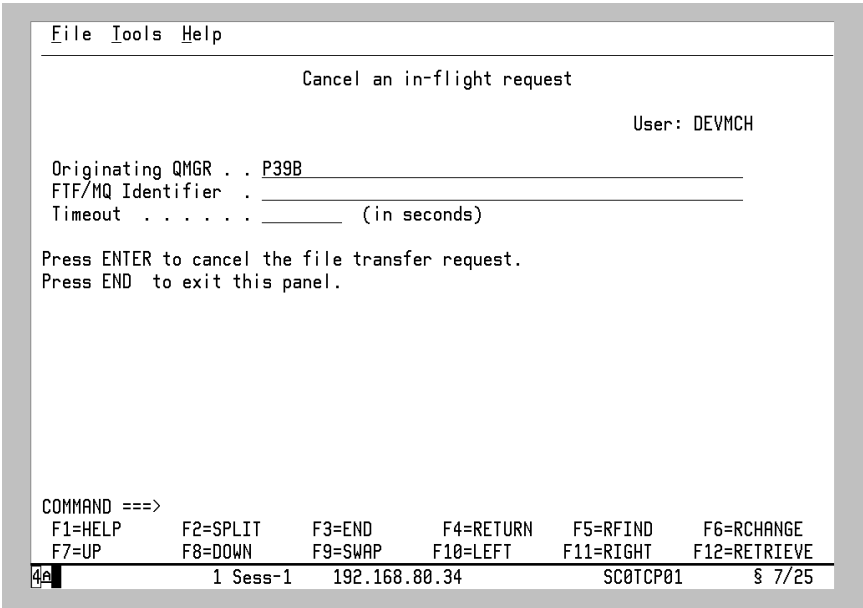
- Status – Last status recorded at the FTF Receiver.
- Status Time – Date and time the listed status was achieved at the FTF Receiver.

**Cancelling a Data-Transfer Request**

The FTFCANCL option on the *Primary options menu* panel allows you to cancel an active data-transfer request. When FTF processes a cancellation request, the transaction is immediately stopped and purged. If you try to cancel a data-transfer request that has already been completed, the cancellation request is ignored.

To cancel a data-transfer request, follow these steps:

1. From the *Primary options menu* panel, type **3** or select **Tools>Cancel Request**. The *Cancel an in-flight request* panel appears. This panel allows you to enter information selecting the data-transfer request to be cancelled.



2. Enter the following information to select the data-transfer request(s) to cancel:

- **Originating QMGR** – Originating queue manager for which data-transfer requests should be canceled.
  - **FTF/MQ Identifier** – FTFID of the data-transfer request to be canceled.
  - **Timeout** – Amount of time, in seconds, to allow for processing the cancel request. **Valid values:** 1-32767
3. Perform one of the following steps:
- Press **[ENTER]** to submit the cancellation request.
  - Press **[END]** to cancel the entries and clear the panel.
  - Press **[PF3]** to return to the *Primary options menu*.

The *Primary options menu* panel appears. Status information about the success or failure of the cancellation appears in the short message area.

## Pinging Components

The FTFPING option on the *Primary options menu* panel allows you to ping specified FTF components. The ping message starts at the local queue manager, the machine from which you are currently working. It is sent to the FTF Manager, as identified by the originating queue manager. The FTF Manager sends it to the FTF Sender, as identified by the source queue manager. The FTF Sender sends it to the FTF Receiver, as identified by the destination queue manager. The FTF Receiver then sends an acknowledgment to the FTF Manager. The acknowledgment appears in the ISPF result set when the ping operation is complete.

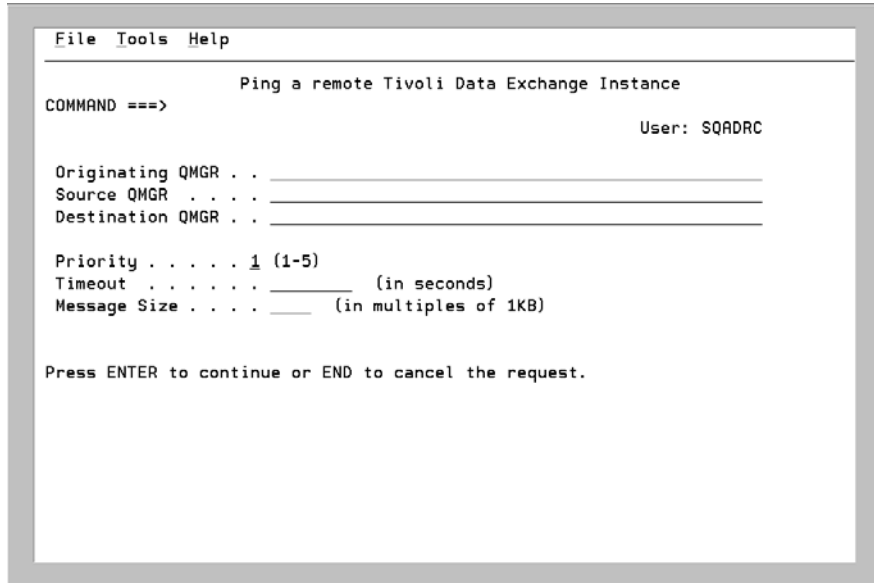
The FTFPING function allows you to ensure that all FTF components involved in a data transfer are available before you start the transfer. You can also use FTFPING with the message size setting to test various message size values until you find the optimal message-size setting.

To ping FTF components, follow these steps:

## Using the ISPF User Interface

### Pinging Components

1. From the *Primary options menu* panel, type **4** or select **Tools>Ping**. The *Ping a remote FTF Instance* panel appears. This panel allows you to specify the FTF components to be pinged.



```
File  Tools  Help

                                Ping a remote Tivoli Data Exchange Instance

COMMAND ===>

                                User: SQADRC

Originating QMGR . . .
Source QMGR . . . .
Destination QMGR . . .

Priority . . . . . 1 (1-5)
Timeout . . . . .      (in seconds)
Message Size . . . .      (in multiples of 1KB)

Press ENTER to continue or END to cancel the request.
```

2. Enter the following information defining the ping operation, as appropriate:
  - **Originating QMGR** – The FTF Manager to be included in the ping. If you do not specify this value, FTF uses the local queue manager (lqm) value, specified in the MQSeries default queue manager.
  - **Source QMGR** – The FTF Sender to be included in the ping.
  - **Destination QMGR** – The FTF Receiver to be included in the ping.
  - **Priority** – The priority applied to the ping. **Valid values:** 1 (highest) – 5 (lowest). **Default value:** 5
  - **Timeout** – The amount of time, in seconds, to allow for processing the ping request. **Valid values:** 1-32767
  - **Message Size** – Sets a message size value that overrides the MQSeries message size value. This value is set in KB. **Valid values:** 1-3906 KB
3. Perform one of the following steps:
  - Press **[ENTER]** to submit the ping request.



- Press **[END]** to cancel the entries and clear the panel.
- Press **[PF3]** to return to the *Primary options menu*.

The *Primary options menu* panel appears. Status information about the success or failure of the cancellation appears in the short message area.

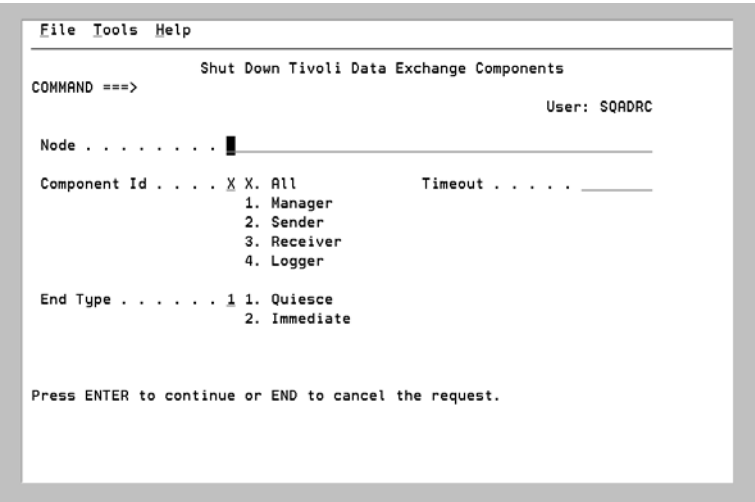
# Shutting Down Components

The FTFEND option on the *Primary options menu* panel allows you to remotely shut down FTF components running on a specified node.

The FTF components being shut down do not process the shutdown message while they are processing a data-transfer request. For example, if the FTF Sender is transferring a large file, it does not read the shutdown message from the control queue until it finishes processing its data. Ending the FTF Sender by other means may cause the failure of the data-transfer request.

To shut down components, follow these steps:

- 1. From the *Primary options menu* panel, type **5** or select **Tools>Shutdown Components**. The *Shut Down FTF Components* panel appears. This panel allows you to specify the FTF components to be shut down.



- 2. In the Node field, enter the node on which you want to shut down FTF components.
- 3. Select the component type(s) to be shut down. The following table lists and describes the component type selections.

Selection	Description
X	Shuts down all specified FTF components
1	Shuts down the specified FTF Manager
2	Shuts down the specified FTF Sender
3	Shuts down the specified FTF Receiver
4	Shuts down the Logger

4. Select an **End Type** value.
  - **Quiesce** – Allows all jobs that are in process to complete in a normal manner, but no new jobs can be started.
  - **Immediate** – Stops all processes at the time the command is entered.
5. Enter the **Timeout** value, which determines the amount of time, in seconds, to allow for processing the shutdown request.
6. Perform one of the following steps:
  - Press **[ENTER]** to submit the shutdown request.
  - Press **[END]** to cancel the entries and clear the panel.
  - Press **[PF3]** to return to the *Primary options menu*.

The *Primary options menu* panel appears. Status information about the success or failure of the cancellation appears in the short message area.

## Exiting the ISPF Panels

To exit the FTF ISPF panels, return to the *Primary Options Menu* panel and perform one of the following steps:

- Type **X**.
- *Or* –
- Select **File>Exit**.



# Using the 5250 User Interface

Tivoli Data Exchange (TDE) includes a 5250 interface that allows you to submit and monitor data-transfer requests in the AS/400 environment. These panels allow you to interactively perform all the tasks the TDE (FTF) commands and APIs perform.

The 5250 interface is useful for FTF administrators monitoring data-transfer requests and submitting ad hoc requests. It is also useful for developers who want to work out the details of a particular data-transfer request before writing code or scripts to perform the transfer. The interface assists in starting and shutting down FTF components.

This chapter describes how to access and use the FTF 5250 interface. It includes the following information:

Section	Page
Accessing the 5250 Interface Commands	168
FTF Request	172
FTF Ping	194
FTF Cancel	196
FTF Status	199
FTF End	203
FTF Stage	206
FTF Configuration	209
Starting an FTF Manager	211
Starting an FTF Sender	214
Starting an FTF Receiver	217
Starting the FTF Logger	221
Logging off FTF	224

## Assumptions

This section makes the following assumptions:

- The 5250 interface has been successfully installed on an IBM computer that has access to the data being transferred and to all FTF components.
- Users will be given appropriate administrative privileges for the functionality in the 5250 panels.

## Accessing the 5250 Interface Commands

Before you can execute FTF commands from the 5250, you must access them. Gaining access involves logging into the AS/400 machine on which the 5250 interface exists and gaining access to the FTF menu.

To access the 5250 interface commands, follow these steps:

1. From the AS/400 log-in panel (displayed below), enter your user ID and password and press **[ENTER]**.

	System . . . . .	:	S1055KKG
	Subsystem . . . . .	:	QBASE
	Display . . . . .	:	QPADEV0010
User . . . . .			_____
Password . . . . .			_____
Program/procedure . . . . .			_____
Menu . . . . .			_____
Current library . . . . .			_____

## Using the 5250 User Interface

### *Accessing the 5250 Interface Commands*

The AS/400 main menu appears. The following figure is an example of an AS/400 main menu. The appearance of your menu can vary.

```
MAIN                                                    System:  S1055KKG
Select one of the following:
    1. User tasks
    2. Office tasks
    4. Files, libraries, and folders
    6. Communications
    8. Problem handling
    9. Display a menu
   10. Information Assistant options
   11. Client Access/400 tasks
   90. Sign off
Selection or command
===>
F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel  F13=Information Assistant
F23=Set initial menu
```

2. To access the main FTF menu, type the following command and press **[ENTER]**:

```
go cmdftftdemq
```

## Using the 5250 User Interface

### *Accessing the 5250 Interface Commands*

The FTF menu, which allows you to run all FTF commands, appears.

The screenshot shows a 5250 terminal window titled 'CMDTDEM0' and 'Tivoli Data Exchange'. It prompts the user to 'Select one of the following:'. The menu is divided into three sections: 'FTF Processing Commands' (items 1-7), 'FTF Component Commands' (items 8-11), and '12. Copyright Information'. To the right of the first two sections, specific command names are listed: FTF, FTFPING, FTFCNCL, FTFSTAT, FTFEND, FTFSTAGE, and FTFCFG for the first section; and STRFTFMGR, STRFTFSDR, STRFTFRCV, and STRFTFLOG for the second section. At the bottom, there is a prompt '====>' followed by a horizontal line for user input.

```
CMDTDEM0                                Tivoli Data Exchange

Select one of the following:

  FTF Processing Commands
    1. FTF Request
    2. FTF Ping
    3. FTF Cancel
    4. FTF Status
    5. FTF End
    6. FTF Stage
    7. FTF Config
  FTF Component Commands
    8. Start FTF Manager
    9. Start FTF Sender
   10. Start FTF Receiver
   11. Start FTF Logger
  12. Copyright Information

Selection or command

====> _____
```

3. To access the panel from which you can execute a FTF command, perform one of the following actions:
  - Type the number of the command you want to execute and press **[ENTER]**. For instance, type **1** to make a FTF data-transfer request.
  - *Or* -
  - Type the command listed with the menu item you want to execute and press **[F4]**. For instance, type **FTF** to access the panels from which you can make a data-transfer request and press **[F4]**. The FTF Request menu appears.

From the FTF menu, you can access panels to run the following commands:

- FTF Request (see page 172)
- FTF Ping (see page 194)
- FTF Cancel (see page 196)
- FTF Status (see page 199)
- FTF End (see page 203)
- FTF Stage (see page 206)
- FTF Config (see page 209)
- Start FTF Manager (see page 211)



- Start FTF Sender (see page 214)
- Start FTF Receiver (see page 217)
- Start FTF Logger (see page 221)

## **Running Commands from the Command Line**

This chapter describes how to use 5250 panels to run FTF commands interactively. You can also run the commands on the command line.

To run commands from the command line, you must use the keyword for each desired parameter. You can view these keywords by pressing **[F11]**. For instance, the keyword for *Local Queue Manager* is LQM.

You can run the command by typing the appropriate command found in the right-hand column of the FTF menu and each desired keyword and pressing **[F4]**. To run the FTF command with an lqm specified, use the following command:

```
FTF LQM(localQMgr)
```

## FTF Request

The FTF Request panel allows you to configure and execute a data-transfer request. Because of the number of possible parameters associated with this command, you must scroll down to enter all the parameters. Each time you press **[Page Down]** to scroll down, more parameters appear. This section describes the FTF Request parameters in groupings based on what appears when you press **[Page Down]** to scroll down.

---

### Note:

When a data-transfer request fails and the file mode for the FTF Receiver is set to “Create” or “Replace,” the Receiver component deletes the file. If the data-transfer request fails before the Receiver starts processing to the target file, cleanup is not required. However, if the mode is set to “Create” or “Replace” and the Receiver has started writing data to the target file when the data-transfer request fails, the Receiver deletes the file.

---

## FTF Request Parameters — Group 1

The following figure displays the first group of parameters used with the FTF command. Type the appropriate parameter values and press **[Page Down]** to see more parameters or **[ENTER]** to execute the command.

FTF Request (FTF)

Type choices, press Enter.

Source Queue Manager . . . . . █

Source File Path . . . . .

Source File Type . . . . .

Source File Data . . . . .

Destination Queue Manager . . .

Destination File Path . . . . .

More...

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display  
F24=More keys

You can enter the following parameters on the first FTF Request panel that appears:

- **Source Queue Manager** – Name of the queue manager where the FTF Sender resides.
- **Source File Path** – Name of the fully qualified path of the source file as it exists on the sending node.
- **Source File Type** – Type of source data to be transferred. This value must be the same as one of the FileTypes values specified in the FTF configuration file. **Recommended value:** AS400PF
- **Source File Data** – A string of values that determines the table name, the record format name, if the data is of a fixed length, and the delimiter character. **Source File Data** takes the following form:

“TBL=*tableName*;RCDFMT=*recordFormatName*;  
FIXLEN=*fixedLength*;DLM=*delimiterChar*”

Where:

- **TBL** *tableName* – Name of the physical or logical file located in lib/*filename(mbrname)*. **Default value:** If *mbrname* is not specified, the first member name is used.

Using the 5250 User Interface  
FTF Request

- **RCDFMT** *recordFormatName* – Source file’s record format.  
**Default value:** If this value is not specified, the first record format is used.
- **FIXLEN** *fixedLength* – Specifies whether the record will have a fixed length at the target file. **Valid values:** Y, N **Default value:** N
- **DLM** *delimiterChar* – Character to be used as the delimiter.  
**Default value:** If this value is not specified, a comma is used.
- **Destination Queue Manager** – Name of the queue manager where the FTF Receiver resides.
- **Destination File Path** – The fully qualified path where the file will be written on the destination node.

FTF Request Parameters – Group 2

The following figure displays the parameters that appear after you press **[Page Down]** the first time. Type the appropriate parameter values and press **[Page Up]** or **[Page Down]** to see more parameters or press **[ENTER]** to execute the command.

The screenshot shows a terminal window titled "FTF Request (FTF)". Inside, it prompts the user to "Type choices, press Enter." and lists several parameters with input lines: "Destination File Type", "Destination File Data", "Local Queue Manager", "Config File", "Config Queue", and "Options File". At the bottom, it lists function key instructions: "F3=Exit", "F4=Prompt", "F5=Refresh", "F12=Cancel", "F13=How to use this display", "F24=More keys", and "More..."

```
FTF Request (FTF)

Type choices, press Enter.

Destination File Type . . . . .
Destination File Data . . . . .
Local Queue Manager . . . . .
Config File . . . . .
Config Queue . . . . .
Options File . . . . .

F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys                                     More...
```

You can enter the following parameters on the FTF Request panel after you scroll down the first time:

- **Destination File Type** – Type of destination file into which the source file is to be transferred. This value must be the same as one of the FileTypes values specified in the FTF configuration file.  
**Recommended value:** AS400PF
- **Destination File Data** – A string of values that determines the table name, the record format name, if the data is of a fixed length, and the delimiter character. **Destination File Data** takes the following form:

“TBL=*tableName*;RCDFMT=*recordFormatName*;  
FIXLEN=*fixedLength*;DLM=*delimiterChar*”

Where:

- **TBL** *tableName* – Name of the physical or logical file located in lib/*filename(mbrname)*. **Default value:** If *mbrname* is not specified, the first member name is used.
- **RCDFMT** *recordFormatName* – Target file’s record format.  
**Default value:** If this value is not specified, the first record format is used.
- **FIXLEN** *fixedLength* – Specifies whether the record will have a fixed length at the target file. **Valid values:** Y, N **Default value:** N
- **DLM** *delimiterChar* – Character to be used as the delimiter.  
**Default value:** If this value is not specified, a comma is used.
- **Local Queue Manager** – Queue manager from which the FTF command is issued. If this value is not specified, the FTF command connects to the default queue manager that is set in the MQSeries configuration.

Otherwise, whenever a command or interface starts up it tries to connect to the local queue manager (lqm). If no lqm value is specified, the command or interface attempts to connect to the specified default queue manager on platforms where MQSeries supports them.

- **Config File** – The fully qualified path and filename for the FTF configuration file. This value is optional if the FTF\_CONFIG\_FILE environment variable is set. If you are using a queue to hold configuration values, do not enter a value in this field.

- **Config Queue** – Queue from which the configuration information is to be retrieved for this FTF instance on this node. The -cq argument points FTF to the queue name rather than to the standard configuration file.
- **Options File** – The fully qualified path and filename of a text file used to contain command-line arguments for the FTF command. In the options file, you can set any of the command-line arguments that can be set for the FTF command. Any values specified on the command line override the values in the options file.

### FTF Request Parameters – Group 3

The following figure displays the parameters that appear after you press **[Page Down]** the second time. Type the appropriate parameter values and press **[Page Up]** or **[Page Down]** to see more parameters or **[ENTER]** to execute the command.

```

                                FTF Request (FTF)

Type choices, press Enter.

Originating Queue Manager . . .

```

---

```

FTF Transaction Label . . . . .
Cancel Mode . . . . .          *DFLT      *ON, *OFF, *DFLT
Pad Records . . . . .          *NOPAD      *PAD, *NOPAD
Padding Character . . . . .          Character value
Record Wrapping . . . . .      *FAIL      *WRAP, *FAIL, *TRUNC
Display Version . . . . .      *NO        *NO, *YES
Seconds to wait for reply . . . . .          Character value
Make Dest Directory . . . . .    *NO        *NO, *YES
Mode of writing file . . . . .    *REPLACE  *APPEND, *NOREPLACE, *REPLACE
File Type . . . . .            *BINARY    *TEXT, *BINARY
Trusted Transfer . . . . .      *NO        *YES, *NO
Enable Compression . . . . .    *NO        *YES, *NO
Transmission Persistent . . . . . *NO      *YES, *NO
Stage Persistent . . . . .      *NO        *YES, *NO

```

---

```

                                More...

F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys

```

You can enter the following parameters on the FTF Request panel after you scroll down the second time:

- **Originating Queue Manager** – The FTF Manager to be included in the FTF command. If you do not specify this value, FTF uses the lqm value.
- **FTF Transaction Label** – The user-identified label that can be associated with the data-transfer request. This can be used as part of status retrieval.
- **Cancel Mode** – Provides a command line override to the preemptive cancel flag in the FTF configuration file. **Valid values:** ON, OFF.
- **Pad Records** – Controls whether records will be padded if there are differences in the length of records. **Valid values:** PAD, NOPAD.
- **Padding Character** – Specifies the character that is used to pad a record. Type \*SPACE if spaces are required. Otherwise, enter the hexadecimal character or the alphanumeric value.

### How Padding Works

FTF padding functionality is governed at the source node (the FTF Sender) and by the characteristics of the data being transferred. No padding decisions are made at the target node (the FTF Receiver). If you request a data transfer in which the source and target nodes both run under operating systems that support record-formatted I/O (such as OS/390), using padding may cause undesired results if you specify different record lengths for your source and target files.

For example, if you send an OS/390 source file with a record length of 80 bytes to an OS/390 target file with a record length of 100 bytes, and if you enable padding, the target file's records are padded up to 80 bytes and contain blanks thereafter.

In addition, if you use the wrap option and the padding option, the padded characters will wrap if the record length of the target file is less than the source file's record length. You can specify padding in the FTF command, the FTF configuration file, and the ISPF panels.

---

**Note:**

The hexadecimal character specified as a padding character is subject to MQSeries data translation rules. The hexadecimal character is converted according to the specification of the target system's code page.

---

- **Record Wrapping** – Indicates how records will be processed when they reach the target file and the records are longer than the target record length. FTF provides the following options:
  - **WRAP** – Wraps records that are of greater length than the target file record length.
  - **TRUNCATE** – Truncates records up to the record length of the target file.
  - **FAIL** – Fails the data-transfer request when the record length exceeds the maximum allowed for the target file.
- **Display Version** – Returns the current FTF version, release, and patch number. Any command entered with this argument ignores all other arguments and returns the version information.
- **Seconds to wait for reply** – Determines the time to wait for a reply from the query operation. Specified in seconds. **Default value:** 300 seconds (5 minutes).
- **Make Dest Directory** – Specifies whether the destination queue manager should make directories. **Valid values:** YES, NO
- **Mode of writing file** – Specifies the disposition of an existing file on the receiving node.
  - **APPEND** – Records are added to an existing file
  - **REPLACE** – Replaces the existing file
  - **NOREPLACE** – The file is not replaced
- **File Type** – Signifies the nature of the data being sent. **Valid values:** BINARY, TEXT **Default value:** BINARY



- **Trusted Transfer** – Sacrifices recovery ability to allow for greater performance. In a trusted data-transfer request, no file recovery is possible. Specifying this argument invokes the FTF trusted option, not the MQSeries trusted option.
- **Enable Compression** – Specifies that the data being sent is compressed using the FTF compression algorithm. **Valid values:** YES, NO
- **Transmission Persistent** – Specifies that the files being transferred are persistent. If you specify this argument, the files still exist after a system or FTF reboot or shutdown. If you select this argument, you increase the recovery ability of FTF, but reduce performance. **Valid values:** YES, NO
- **Stage Persistent** – Specifies that the messages in the staging queue are persistent. If you specify this argument, the messages still exist after a system or FTF reboot or shutdown. If you select this argument, you increase the recovery ability of FTF, but reduce performance. **Valid values:** YES, NO

## FTF Request Parameters – Group 4

The following figure displays the parameters that appear after you press [Page Down] the third time. Type the appropriate parameter values and press [Page Up] or [Page Down] to see more parameters or press [ENTER] to execute the command.

FTF Request (FTF)

Type choices, press Enter.

Data Pool Name . . . . .		
Override Max Msg Size . . . . .		Character value
Transfer Priority . . . . .		Character value
Stage The Source . . . . .	*NO	*YES, *NO
Stage Only . . . . .	*NO	*YES, *NO
From Stage . . . . .	*NO	*YES, *NO
FTF Transfer ID . . . . .		
Delete The Source . . . . .	*NO	*YES, *NO
Immediate Transfer Mode . . . . .	*NO	*YES, *NO
DBCS Processing . . . . .	*NO	*YES, *NO
Expiry Time . . . . .		Character value
Reply Queue . . . . .		
Reply Queue Manager . . . . .		
More...		
F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display		
F24=More keys		

### Specifying an Esoteric Unit Name

To specify an esoteric name for the OS/390 UNIT value, follow these steps:

- Do not set a value in the MVSVOLUME stanza.
- Specify the unit value in the *MVS UNIT* field, or in the MVSUNITNAME stanza in the FTF configuration file on either the FTF Sender or the FTF Receiver.

You can enter the following parameters on the FTF Request panel after you scroll down the third time:

- **Data Pool Name** – Name of the data pool used for transferring data from the FTF Sender to the FTF Receiver. If this value is not specified, FTF uses the default pool specified in the FTF configuration file. For this option to function, the specified pool must be defined in the FTF configuration file.

- **Override Max Msg Size** – Allows you to set a message size value to override the MQSeries message size value. **Valid values:** 1-3906 KB (3.9 MB) **Default value:** 512
- **Transfer Priority** – Priority applied to the data-transfer request. **Valid values:** 1 (highest) – 5 (lowest) **Default value:** 5
- **Stage The Source** – Enables the data messages that make up the data being transferred to be stored in a staging queue until the data-transfer request is complete. Staging is appropriate if the data being transferred will not permanently exist on the sender. It allows the data-transfer request to resend the message from the staging queue. **Valid values:** YES, NO
- **Stage Only** – Places the data messages on the staging queue, but does not send them. If you specify this value, you cannot specify a dqm or destination file value. **Valid values:** YES, NO
- **From Stage** – Indicates that the source file specified in the Source Path File should be retrieved from the staging queue rather than from an actual source file. **Valid values:** YES, NO

## Specifying the Staging Queue

The staging queue is specified by the *SenderStageControlQueue* property in the FTF configuration file. The default value for this queue is FTFSDR.STAGE.CONTROL. To override this default, change the property value to the appropriate queue name and restart FTF.

---

### Note:

If you specify a filename for the transfer from the staging queue and multiple entries have the same filename, the file related to the first matching entry found is sent. If you need to specify a specific instance of a file in the staging queue, use that file's FTFID.

---

- **FTF Transfer ID** – Indicates the FTFID of the data-transfer request that should be sent from the staging queue. Use this option only if you are sending a file from the staging queue and specifying From Stage, and you have not specified a source filename. The FTFID value is automatically generated when the data-transfer request is submitted.
- **Delete the Source** – Indicates that the source data is to be deleted once the data-transfer request is completed. **Valid values:** YES, NO
- **Immediate Transfer Mode**– Gives you the ability to issue an FTF data-transfer request that is processed synchronously between the Sender and Receiver rather than the normal asynchronous mode. Using this argument differentiates an immediate data transfer from a regular FTF transfer. Using FTF immediate, the FTF Receiver begins to process the data-transfer request as soon as it gets the request instead of waiting for all of the data to arrive. Since the receiver processes the data messages as they arrive, the queue storage required to transfer a large amount of data is significantly reduced. Normal FTF data transfers require that the Receiver have all related data messages before writing the data to the disk. For example, using FTF in normal mode when processing very large data transfers such as 1 gigabyte, the receiving node needs to have 1 gigabyte for queue storage and 1 gigabyte to store the data. With FTF immediate, the storage space on the queue storage is not required. **Valid values:** YES, NO.

### **Immediate Works with Connectors**

The connector library accepts the immediate option. Immediate file transfers and immediate connector transfers both operate in the same manner.

---

## Considerations with Immediate

An immediate transfer is a synchronous transaction. If either the Sender or the Receiver goes down or if the connection between the Sender and Receiver is lost during processing, the transaction times out and fails. For example, if the Sender is processing an immediate transfer and the Receiver is down, the Sender times out and fails the transaction. Likewise, if the Sender is processing an immediate transfer and it goes down, then the Receiver times out and fails the transaction.

If the Receiver starts processing an immediate transfer and then goes down, it cannot recover the request. The request fails. The Sender, however, can recover if it goes down during processing of an immediate transfer as long as it is restarted prior to the Receiver timing out and failing the transfer.

---

---

## Note:

When either the Sender or the Receiver times out and fails a transaction, a cancel message is sent to notify the other side to stop processing and to clean up any messages. This is when the failed transaction appears to be cancelled. You should look at the detailed status for the transaction to determine the reason for failure. The status logs reflect two transactions:

1. A failed transaction message when the Sender recognized that the Receiver was not responding.
  2. A cancelled transaction message when the Receiver comes back up and recognizes the cancellation message from the Sender.
-

- **DBCS Processing** – Specifies the option to scan the source file and split each datablock at the point where double-byte validity is maintained. Using this option ensures that there is no truncation of a double-byte character when the specified message size does not span the entire character. If the DBCS option is used with the wrap option in a data-transfer request, the results can be unpredictable. **Valid values:** YES, NO
- **Expiry Time** – Determines the time period, measured in minutes, after which the data-transfer request expires. If the expiration duration is exceeded, the request is terminated and the FTFRCI\_REQUEST\_EXPIRED message is returned.
- **Reply Queue** – Name of the queue to which reply messages are to be routed.
- **Reply Queue Manager** – Name of the queue manager to which reply messages are to be routed.

FTF Request Parameters – Group 5

The following figure displays the parameters that appear after you press **[Page Down]** the fourth time. Type the appropriate parameter values and press **[Page Up]** or **[Page Down]** to see more parameters or press **[ENTER]** to execute the command.

FTF Request (FTF)

Type choices, press Enter.

Notification Status . . . . .

Notification Type . . . . .

Notification Data . . . . .

AS400 File Options:

AS400 File Type . . . . .

AS400 Create Library . . . . .

AS400 Library Aux Storage Pool . . . . .

AS400 File Aux Storage Pool . . . . .

AS400 Library Text . . . . .

AS400 File Text . . . . .

AS400 CCSID . . . . .

AS400 Record Length . . . . .

F3=Exit

F4=Prompt

F5=Refresh

F12=Cancel

F13=How to use this display

F24=More keys

'', \*FAILURE, \*SUCCESS...

\*DFLT, \*SAVE, \*SRCPF

\*NO, \*YES

Character value

Character value

Character value

Character value

More...

You can enter the following parameters on the FTF Request panel after you scroll down the fourth time:

- **Notification Status** – Defines when a notification message will be sent to the Notify Queue. The notification is sent if the transaction's status matches the status specified in the configuration file. If you specify this property, you must also specify Notification Data and Type. **Valid values:** Success, Failure, Nonsuccess (includes failed, cancelled, expired)
- **Notification Type** – Specifies the user-defined method that will be used to deliver a notification message based on a transaction's status. Examples of user-defined values are: EMAIL, PAGER, FAX, WTO. If you specify this property, you must also specify Notification Data and Status.
- **Notification Data** – Specifies user-defined data to aid in notification, such as e-mail, pager, or fax information, that will be used to deliver a notification message based on a transaction's status. If you specify this property, you must also specify Notification Status and Type.
- **AS400 File Type** – The value entered specifies the type of file.
  - **\*DFLT** – Defaults to the value entered in the configuration table
  - **\*SAVE** – Specifies an AS/400 Save file
  - **\*SRCPF** – Specifies an AS/400 source physical file
- **AS400 Create Library** – Specifies that FTF is to create the specified library if it does not exist. **Valid Values:** YES, NO.
- **AS400 Library Aux Storage Pool** – Specifies the library auxiliary pool for a library that FTF creates for a data-transfer request.
- **AS400 File Aux Storage Pool** – Specifies the library auxiliary pool for a file that FTF creates for a data-transfer request.
- **AS400 Library Text** – Specifies the library description for a library that FTF creates for a data-transfer request.
- **AS400 File Text** – Specifies the file description for a library that FTF creates for a data-transfer request.
- **AS400 CCSID** – The CCSID is used as the identifier for the data-transfer request. If CCSID is not specified, FTF will use the CCSID of the job.
- **AS400 Record Length** – Determines the logical record length in bytes for the target file on OS/390. **Valid values:** 1-327606

**Note:**

On the AS/400, if the record length of the data being sent is over 80 bytes, then the Record Length parameter should always be 12 greater than the actual record length. For example, if you send a file with the OS/390 lrecl equal to 1024, then you need to set the record length to 1036.

**FTF Request Parameters – Group 6**

The following figure displays the parameters that appear after you press [Page Down] the fifth time. Type the appropriate parameter values and press [Page Up] or [Page Down] to see more parameters or press [ENTER] to execute the command.

FTF Request (FTF)

Type choices, press Enter.

MVS File Options:

MVS Org . . . . .

MVS RECFMT . . . . .

MVS LRECL . . . . .

MVS BLKSIZE . . . . .

MVS UNIT . . . . .

MVS VOLSER . . . . .

MVS ALCUNIT . . . . .

MVS Primary Allocation . . .

MVS Secondary Allocation . .

MVS GDG Model . . . . .

MVS BUFNO . . . . .

Mgr Pre Exit . . . . .

Character value

Character value

Character value

Character value

Character value

Character value

Character value

Character value

Character value

Character value

Character value

More...

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display

F24=More keys

You can enter the following parameters on the FTF Request panel after you scroll down the fifth time:

- **MVS Org** – File organization of the target file on OS/390. **Valid values:** Physical Sequential (PS), Partitioned Data Set (PDS)



- **MVS RECFMT** – Record format for the target file on OS/390. **Valid values:** F (fixed), V (variable), FB (fixed block), and VB (variable block)
- **MVS LRECL** – Logical record length in bytes for the target file on OS/390. **Valid values:** 1-32760
- **MVS BLKSIZE** – Block size in bytes for the target file on OS/390. Specifying a block size of 0 enables the system to choose the optimum block size for the data set during allocation. If the record format is Fixed Block (FB), the block size in the blksize argument must be a multiple of the logical record length, the lrecl parameter. When the record format is Variable Block (VB), the blksize value must be at least four bytes greater than the lrecl value. **Valid values:** 0-32760
- **MVS UNIT** – Unit name for the target file. This argument's value is installation-dependent. Obtain it from your OS/390 administrator.
- **MVS VOLSER** – Volume serial number for the target file. This argument's value is installation-dependent. Obtain it from your OS/390 administrator.
- **MVS ALCUNIT** – Allocation unit used for the target on OS/390. **Valid values:** CYL (cylinder), BLK (block), TRK (track)
- **MVS Primary Allocation** – Number of primary allocation units required.
- **MVS Secondary Allocation** – Number of secondary allocation units required.
- **MVS GDG Model** – A model data set for Generation Data Group (GDG) allocation. Consult your OS/390 administrator for the available model data sets.
- **MVS BUFNO** – Number of internal buffers that are to be used when processing data transfers. The throughput of a FTF data transfer is governed by a combination of the block size of the data being transferred and the number of buffers that are allocated for transfer in the bufno argument. **Valid Values:** 1 - 255
- **Mgr Pre Exit** – Name of the service program that is to be used for the manager preprocess exit. This field is mandatory if you want to execute a manager preprocess exit.

**FTF Request Parameters – Group 7**

The following figure displays the parameters that appear after you press **[Page Down]** the sixth time. Type the appropriate parameter values and press **[Page Up]** or **[Page Down]** to see more parameters or press **[ENTER]** to execute the command.

The screenshot shows a terminal window titled "FTF Request (FTF)". Inside, the text "Type choices, press Enter." is at the top. Below it are four input fields, each preceded by a label and a series of dots: "Mgr Pre Entry", "Mgr Pre Data", "Mgr Post Exit", and "Mgr Post Entry". The first field has a cursor. At the bottom right, there is a "More..." link. At the bottom left, a legend lists function keys: F3=Exit, F4=Prompt, F5=Refresh, F12=Cancel, F13=How to use this display, and F24=More keys.

You can enter the following parameters on the FTF Request panel after you scroll down the sixth time:

- **Mgr Pre Entry** – Functional entry point within the service program specified in the **Mgr Pre Exit**.
- **Mgr Pre Data** – Data passed to the exit module. If it is necessary to specify an exitdata value of spaces, then enclose the spaces in single quotes.
- **Mgr Post Exit** – Name of the service program that is to be used for the manager postprocess exit. This field is mandatory if you want to execute a manager postprocess exit.
- **Mgr Post Entry** – Functional entry point within the service program specified in the **Mgr Post Exit**.

## FTF Request Parameters – Group 8

The following figure displays the parameters that appear after you press **[Page Down]** the seventh time. Type the appropriate parameter values and press **[Page Up]** or **[Page Down]** to see more parameters or press **[ENTER]** to execute the command.

FTF Request (FTF)

Type choices, press Enter.

Mgr Post Data . . . . .

Sender Pre Exit . . . . .

Sender Pre Entry . . . . .

Sender Pre Data . . . . .

More...

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display  
F24=More keys

You can enter the following parameters on the FTF Request panel after you scroll down the seventh time:

- **Mgr Post Data** – Data passed to the exit module. If it is necessary to specify an exitdata value of spaces, then enclose the spaces in single quotes.
- **Sender Pre Exit** – Name of the service program that is to be used for the sender preprocess exit. This field is mandatory if you want to execute a sender preprocess exit.
- **Sender Pre Entry** – Functional entry point within the service program specified in the **Sender Pre Exit**.
- **Sender Pre Data** – Data passed to the exit module. If it is necessary to specify an exitdata value of spaces, then enclose the spaces in single quotes.

FTF Request Parameters – Group 9

The following figure displays the parameters that appear after you press **[Page Down]** the eighth time. Type the appropriate parameter values and press **[Page Up]** or **[Page Down]** to see more parameters or press **[ENTER]** to execute the command.

FTF Request (FTF)

Type choices, press Enter.

Sender Post Exit . . . . .

Sender Post Entry . . . . .

Sender Post Data . . . . .

Receiver Pre Exit . . . . .

More...

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display  
F24=More keys

You can enter the following parameters on the FTF Request panel after you scroll down the eighth time:

- **Sender Post Exit** – Name of the service program that is to be used for the sender postprocess exit. This field is mandatory if you want to execute a sender postprocess exit.
- **Sender Post Entry** – Functional entry point within the service program specified in the **Sender Post Exit**.
- **Sender Post Data** – Data passed to the exit module. If it is necessary to specify an exitdata value of spaces, then enclose the spaces in single quotes.
- **Receiver Pre Exit** – Name of the service program that is to be used for the receiver preprocess exit. This field is mandatory if you want to execute a receiver preprocess exit.

## FTF Request Parameters – Group 10

The following figure displays the parameters that appear after you press **[Page Down]** the ninth time. Type the appropriate parameter values and press **[Page Up]** or **[Page Down]** to see more parameters or press **[ENTER]** to execute the command.

FTF Request (FTF)

Type choices, press Enter.

Receiver Pre Entry . . . . . \_\_\_\_\_

Receiver Pre Data . . . . . \_\_\_\_\_

Receiver Post Exit . . . . . \_\_\_\_\_

Receiver Post Entry . . . . . \_\_\_\_\_

More...

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display  
F24=More keys

You can enter the following parameters on the FTF Request panel after you scroll down the ninth time:

- **Receiver Pre Entry** – Functional entry point within the service program specified in the **Receiver Pre Exit**.
- **Receiver Pre Data** – Data passed to the exit module. If it is necessary to specify an exitdata value of spaces, then enclose the spaces in single quotes.
- **Receiver Post Exit** – Name of the service program that is to be used for the Receiver postprocess exit. This field is mandatory if you want to execute a Receiver postprocess exit.
- **Receiver Post Entry** – Specifies the functional entry point within the service program specified in the **Receiver Post Exit**.

FTF Request Parameters – Group II

The following figure displays the parameters that appear after you press [Page Down] the tenth time. Type the appropriate parameter values and press [Page Up] or [Page Down] to see more parameters or press [ENTER] to execute the command.

FTF Request (FTF)

Type choices, press Enter.

Receiver Post Data . . . . .

Sender Connector Exit . . . . .

Sender Connector Entry . . . . .

Sender Connector Data . . . . .

More...

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display  
F24=More keys

You can enter the following parameters on the FTF Request panel after you scroll down the tenth time:

- **Receiver Post Data** – Data passed to the exit module. If it is necessary to specify an exitdata value of spaces, then enclose the spaces in single quotes.
- **Sender Connector Exit** – Name of the service program for the sender connector exit. This entry is required if the sender connector exit is to be used.
- **Sender Connector Entry** – Functional entry point within the service program specified in the **Sender Connector Exit**.
- **Sender Connector Data** – Data passed to the connector module. If it is necessary to specify an exitdata value of spaces, then enclose the spaces in single quotes.

## FTF Request Parameters – Group 12

The following figure displays the parameters that appear after you press **[Page Down]** the eleventh time. Type the appropriate parameter values and press **[Page Up]** to see more parameters or press **[ENTER]** to execute the command.

FTF Request (FTF)

Type choices, press Enter.

**Receiver Connector Exit** . . . . . \_\_\_\_\_

\_\_\_\_\_

**Receiver Connector Entry** . . . . . \_\_\_\_\_

\_\_\_\_\_

**Receiver Connector Data** . . . . . \_\_\_\_\_

\_\_\_\_\_

Bottom

F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display  
 F24=More keys

You can enter the following parameters on the FTF Request panel after you scroll down the eleventh time:

- **Receiver Connector Exit** – Name of the service program for the receiver connector exit. This entry is required if the receiver connector exit is to be used.
- **Receiver Connector Entry** – Functional entry point within the service program specified in the **Receiver Connector Exit**.
- **Receiver Connector Data** – Data passed to the connector module. If it is necessary to specify an exitdata value of spaces, then enclose the spaces in single quotes.

# FTF Ping

The Ping FTF panel (displayed below) allows you to ping specified FTF components. The ping message starts at the lqm, the machine from which you are currently working. It is sent to the FTF Manager, as identified by the oqm. The FTF Manager sends it to the FTF Sender, as identified by the sqm. The FTF Sender sends it to the FTF Receiver, as identified by the dqm. The FTF Receiver then sends an acknowledgment to the FTF Manager. The acknowledgment appears in the ISPF result set when the ping operation is complete.

By using FTF Ping, you can ensure that all FTF components involved in a data transfer are available before you start the transfer. You can also use FTF Ping with the message size setting to test various message size values until you find the optimal message size setting. This section describes the FTF Ping parameters in groupings based on what appears when you press [Page Down] to scroll down.

The following figure displays the first group of parameters that appear when you use the FTF Ping command. Type the appropriate parameter values and press [Page Down] to see more parameters or press [ENTER] to execute the command.

Ping FTF (FTFPING)

Type choices, press Enter.

Local Queue Manager . . . . .

Originating Queue Manager . . .

Source Queue Manager . . . . .

Destination Queue Manager . . .

Config File . . . . .

Config Queue . . . . .

Timeout Value (sec) . . . . .

Message Size . . . . .

Character value

Character value

More...

F3=Exit

F4=Prompt

F5=Refresh

F12=Cancel

F13=How to use this display

F24=More keys



You can enter the following arguments on the Ping FTF panel:

- **Local Queue Manager** – Queue manager from which the FTF Ping command is issued. If no lqm value is specified, the FTF Ping command connects to the MQSeries default queue manager. Otherwise, whenever a command or interfaces starts up it always tries to connect to the lqm. If an lqm value is not specified, the interface attempts to connect to the specified default queue manager on platforms where MQSeries supports them.
- **Originating Queue Manager** – The FTF Manager to be included in the ping. If you do not specify this value, FTF uses the lqm value.
- **Source Queue Manager** – Queue manager where the FTF Sender resides.
- **Destination Queue Manager** – Queue manager where the FTF Receiver resides.
- **Config File** – The fully qualified path and filename for the FTF configuration file. This value is optional if the FTF\_CONFIG\_FILE environment variable is set. If you are using a queue to hold configuration values, do not enter a value in this field.
- **Config Queue** – Queue from which the configuration information is to be retrieved for this FTF instance on this node. The -cq argument points FTF to the queue name rather than to the standard configuration file.
- **Timeout Value (sec)** – Amount of time, in seconds, to allow for processing the ping request. **Valid values:** 1-32767
- **Message Size** – Allows you to set a message size value to override the MQSeries message size value. **Valid values:** 1 Byte - 4 MB **Default value:** 256 Bytes

## FTF Ping Parameters – Group 2

The following figure displays the parameters that appear after you press **[Page Down]** once. Type the appropriate parameter values and press **[Page Up]** to view more parameters or press **[ENTER]** to execute the command.

Ping FTF (FTFPING)

Type choices, press Enter.

Priority 1..5 . . . . .

Character value

Options File . . . . .

Display Version . . . . .

\*NO

\*NO, \*YES

F3=Exit

F4=Prompt

F5=Refresh

F12=Cancel

F13=How to use this display

F24=More keys

Bottom

You can enter the following parameters after you scroll down once:

- **Priority 1..5** – Priority applied to the data-transfer request. **Valid values:** 1 (highest) – 5 (lowest) **Default value:** 5
- **Options File** – The fully qualified path and filename of a text file used to contain command-line arguments for the FTF Ping command. In the options file, you can set any of the command-line arguments that can be set for the FTF Ping command. Any values specified on the command line override the values in the options file.
- **Display Version** – Returns the current FTF version, release, and patch number. Any command entered with this argument ignores all other arguments and returns the version information.

FTF Cancel

The FTF Cancel panel allows you to construct commands that cancel a message before it is delivered to other components of the system. This section describes the FTF Cancel parameters in groupings based on what appears when you press [Page Down] to scroll down.

## Cancel FTF Request – Group 1

The following figure displays the first group of parameters used with the FTF Cancel command. Type the appropriate parameter values and press **[Page Down]** for more parameters or press **[ENTER]** to execute the command.

```

Cancel FTF Request (FTFCNCL)

Type choices, press Enter.

FTF Transaction Identifier . . . _____
Local Queue Manager . . . . . _____
Config File . . . . . _____

Config Queue . . . . . _____
Originating Queue Manager . . . _____
Options File . . . . . _____

More...

F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
  
```

You can enter the following arguments on the FTF Cancel panel:

- **FTF Transaction Identifier** – The alphanumeric identifier associated with the data-transfer request you want to cancel.
- **Local Queue Manager** – Queue manager from which the FTF Cancel command is issued. If no lqm value is specified, the FTF Cancel command connects to the MQSeries default queue manager. Otherwise, whenever a command or interfaces starts up it always tries to connect to the lqm. If an lqm value is not specified, the interface attempts to connect to the specified default queue manager on platforms where MQSeries supports them.
- **Config File** – The fully qualified path and filename for the FTF configuration file. This value is optional if the FTF\_CONFIG\_FILE environment variable is set. If you are using a queue to hold configuration values, do not enter a value in this field.

## Using the 5250 User Interface

### *FTF Cancel*

- **Config Queue** – Queue from which the configuration information is to be retrieved for this FTF instance on this node. The -cq argument points FTF to the queue name rather than to the standard configuration file.
- **Originating Queue Manager** – The FTF Manager to be included in the cancel command. If you do not specify this value, FTF uses the lqm value.
- **Options File** – The fully qualified path and filename of a text file used to contain command-line arguments for the FTF Cancel command. In the options file, you can set any of the command-line arguments that can be set for the FTF Cancel command. Any values specified on the command line override the values in the options file.

## FTF Cancel Parameters – Group 2

The following figure displays the parameters that appear after you press **[Page Down]** one time. Type the appropriate parameter values and press **[Page Up]** to view more parameters or press **[ENTER]** to execute the command.

```
Cancel FTF Request (FTFCNCL)

Type choices, press Enter.

Display Version . . . . . NO *NO, *YES

Bottom

F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

You can enter the following parameter after you scroll down once:

- **Display Version** – Returns the current FTF version, release, and patch number. Any command entered with this argument ignores all other arguments and returns the version information.

## FTF Status

The FTF Status panel allows you to construct a command that checks the status of other components in the system. This section describes the FTF Status parameters in groupings based on what appears when you press **[Page Down]** to scroll down.

### FTF Status Parameters – Group 1

The following figure displays the first group of parameters used with the FTF Status command. Type the appropriate parameter values and press **[Page Down]** to see more parameters or press **[ENTER]** to execute the command.

FTF Status (FTFSTAT)

Type choices, press Enter.

Local Queue Manager . . . . . █

Config File . . . . .

---

Config Queue . . . . .

---

Purge Status Queues . . . . . \*NO      \*NO, \*YES

Format . . . . . \*TERSE      \*TERSE, \*LONG, \*DETAIL

Include orphans . . . . . \*NO      \*NO, \*YES

FTF Transfer ID . . . . .

---

User label . . . . .

Earliest date of transfer . . .

Latest date of transfer . . .

More...

F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display

F24=More keys

## Using the 5250 User Interface

### *FTF Status*

You can enter the following arguments on the FTF Status panel that appears before you scroll down:

- **Local Queue Manager** – Queue manager from which the FTF Status command is issued. If no lqm value is specified, the FTF Status command connects to the MQSeries default queue manager. Otherwise, whenever a command or interface starts up it always tries to connect to the lqm. If an lqm value is not specified, the interface attempts to connect to the specified default queue manager on platforms where MQSeries supports them.
- **Config File** – The fully qualified path and filename for the FTF configuration file. This value is optional if the FTF\_CONFIG\_FILE environment variable is set. If you are using a queue to hold configuration values, do not enter a value in this field.
- **Config Queue** – Queue from which the configuration information is to be retrieved for this FTF instance on this node. The -cq argument points FTF to the queue name rather than to the standard configuration file.
- **Purge Status Queues** – Permanently removes all of the entries from the status queue if they match all of the filter options. **Valid Values:** YES, NO
- **Format** – The output type for displaying status messages. **Valid Values:** \*SHORT, \*TERSE, \*LONG, \*DETAIL
- **Include orphans** – Forces processing to consider entries in the detail queue that do not have corresponding entries in the control queue. **Valid Values:** YES, NO
- **FTF Transfer ID** – FTFID of the data-transfer request for which status information is returned.
- **User label** – User-specified label for which status information is returned.
- **Earliest date of transfer** – Beginning of the date range for which status information is returned. If you specify the beginning of the date range, but no end, FTF returns all status information logged since the specified date. **Format:** YYYYMMDDhhmmss, where YYYY is the four-digit year, MM is the two-digit month, DD is the day, hh is the hour (24-hour format), mm is the minute, and ss is the second.

- **Latest date of transfer** – End of the date range for which status information is returned. If you specify the end of the date range, but no beginning, FTF returns all status information logged until the specified date. **Format:** YYYYMMDDhhmmss, where YYYY is the four-digit year, MM is the two-digit month, DD is the day, hh is the hour (24-hour format), mm is the minute, and ss is the second.

## FTF Status – Group 2

The following figure displays the parameters that appear after you press **[Page Down]** the first time. Type the appropriate parameter values and press **[Page Up]** or **[Page Down]** to see more parameters or press **[ENTER]** to execute the command.

The screenshot shows a terminal window titled "FTF Status (FTFSTAT)". The prompt "Type choices, press Enter." is at the top. Below it are several lines of text, each followed by a horizontal line for input. The first line is "Status . . . . .", followed by a cursor and the word "ALL", and then "\*ALL, \*COMP, \*FAIL...". The other lines are "Originating Queue Manager . . .", "Requesting Queue Manager . . .", "Source Queue Manager . . . . .", "Source fullpath filename . . .", and "Destination Queue Manager . . .". At the bottom, there is a "More..." prompt and a list of function key shortcuts: F3=Exit, F4=Prompt, F5=Refresh, F12=Cancel, F13=How to use this display, and F24=More keys.

```

FTF Status (FTFSTAT)

Type choices, press Enter.

Status . . . . . ALL *ALL, *COMP, *FAIL...
Originating Queue Manager . . . 
Requesting Queue Manager . . . 
Source Queue Manager . . . . . 
Source fullpath filename . . . 
Destination Queue Manager . . . 

More...
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
  
```

You can enter the following parameters after you scroll down once:

- **Status** – Allows you to specify the type of messages that are returned during a status request. **Valid Values:**
  - **\*COMP** – Include only completed requests
  - **\*FAIL** – Include only failed requests
  - **\*ACTIVE** – Include only active requests
  - **\*CANC** – Include only cancelled requests
  - **\*EXP** – Include only expired requests
  - **\*ALL** – Include status information for requests with any status type
- **Originating Queue Manager** – The FTF Manager for which status information is returned.
- **Requesting Queue Manager** – Name of the queue manager where the data-transfer request is entered.
- **Source Queue Manager** – The FTF Sender for which status information is returned.
- **Source fullpath filename** – The fully qualified source path and filename for which status information is returned.
- **Destination Queue Manager** – Name of the FTF Receiver for which status information is returned.

## **FTF Status Parameters – Group 3**

The following figure displays the parameters that appear after you press [**Page Down**] the second time. Type the appropriate parameter values and press [**Page Up**] to see more parameters or [**ENTER**] to execute the command.



```

                                FTF Status (FTFSTAT)

Type choices, press Enter.

Destination fullpath filename . █

Ignore case sensitivity . . . . *NO          *NO, *YES
File containing options . . . .

Display Version . . . . . *NO          *NO, *YES

                                Bottom
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys

```

- **Destination fullpath filename** – The fully qualified target path and filename for which status information is returned.
- **Ignore case sensitivity** – Ignores case sensitivity for the source and destination path and filename.
- **File containing options** – The fully qualified path and filename of a text file used to contain command-line arguments for the FTF Status command. In the options file, you can set any of the command-line arguments that can be set for the FTF Status command. Any values specified on the command line override the values in the options file.
- **Display Version** – Returns the current FTF version, release, and patch number. Any command entered with this argument ignores all other arguments and returns the version information.

## FTF End

The End FTF Components panel allows you to construct a command that stops the operation of components on the system. This section describes the FTF End parameters.

Using the 5250 User Interface

FTF End

The FTF components being shut down do not process the shutdown message while they are processing a data-transfer request. For example, if the FTF Sender is transferring a large file, it does not read the shutdown message from the control queue until it finishes processing its data. Ending the FTF Sender by other means may cause the failure of the data-transfer request.

Type the appropriate parameter values and press [ENTER] to execute the command.

End FTF Component(s) (FTFEND)

Type choices, press Enter.

Local Queue Manager . . . . .

Config File . . . . .

Config Queue . . . . .

FTF Components Host Node . . . .

Terminate With Pending Work . .

Complete All Work . . . . .

Seconds to wait for response . .

Component To End . . . . .

Display Version . . . . .

\*NO

\*NO

\*ALL

\*NO

\*NO, \*YES

\*NO, \*YES

Character value

\*ALL, \*FTFMGR, \*FTFSDR...

\*NO, \*YES

F3=Exit

F4=Prompt

F5=Refresh

F12=Cancel

F13=How to use this display

F24=More keys

Bottom

You can enter values for the following arguments on the End FTF Components panel:

- Local Queue Manager** – Queue manager from which the FTF End command is issued. If no lqm value is specified, the FTF End command connects to the MQSeries default queue manager. Otherwise, whenever a command or interface starts up it always tries to connect to the lqm. If an lqm value is not specified, the interface attempts to connect to the specified default queue manager on platforms where MQSeries supports them.
- Config File** – The fully qualified path and filename for the FTF configuration file. This value is optional if the FTF\_CONFIG\_FILE environment variable is set. If you are using a queue to hold configuration values, do not enter a value in this field.

204

Tivoli Data Exchange User's Guide

- **Config Queue** – Queue from which the configuration information is to be retrieved for this FTF instance on this node. The -cq argument points FTF to the queue name rather than to the standard configuration file.
- **FTF Components Host Node** – Name of the computer to which the shutdown request is sent.
- **Terminate With Pending Work** – Shuts down the FTF components without regard for the pending work. **Valid Values:** YES, NO
- **Complete All Work** – Shuts down the FTF components after the work on all has been completed. **Valid Values:** YES, NO
- **Seconds to wait for response** – Time to wait for a reply to the FTF End operation. This value does not apply to the purge operation. Specified in seconds. **Default value:** 300 seconds (5 minutes).
- **Component To End** – The component to shut down of all of the components. **Valid Values:**
  - \*ALL - All components
  - \*FTFMGR - FTF Manager
  - \*FTFSDR - FTF Sender
  - \*FTFRCV - FTF Receiver
- **Display Version** – Returns the current FTF version, release, and patch number. Any command entered with this argument ignores all other arguments and returns the version information. **Valid Values:** YES, NO

---

### Note:

When you submit an FTFEND request to shut down FTF component(s) on the lqm, a shutdown message is placed on the queue defined as the FTFLogQueue in the FTF configuration file. If this queue is defined as a remote queue, the FTFLOG program running on the remote queue manager is shut down. This shutdown may be an issue if you are centralizing logging and using the FTFLOG program to write the messages to log files.

---

FTF Stage

The Stage FTF panel allows you to query the staging queue for a list of its contents or to purge specified contents. This section describes the FTF Stage parameters in groupings based on what appears when you press [Page Down] to scroll down.

FTF Stage Parameters – Group 1

The following figure displays the first group of parameters used with the Stage FTF panel. Type the appropriate parameter values and press [Page Down] to see more parameters or press [ENTER] to execute the command.

Stage FTF (FTFSTAGE)

Type choices, press Enter.

Local Queue Manager . . . . .

Originating Queue Manager . . .

Source Queue Manager . . . . .

Config File . . . . .

Config Queue . . . . .

Query . . . . . \*NO \*NO, \*YES

Purge . . . . . \*NO \*NO, \*YES

All . . . . . \*NO \*NO, \*YES

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display

F24=More keys

More...

You can enter the following parameters on the Stage FTF panel that appears before you scroll down.

- **Local Queue Manager** – Queue manager from which the FTF Stage command is issued. If no lqm value is specified, the FTF Stage command connects to the MQSeries default queue manager. Otherwise, whenever a command or interface starts up it always tries to connect to the lqm. If an lqm value is not specified, the interface attempts to connect to the specified default queue manager on platforms where MQSeries supports them.
- **Originating Queue Manager** – The FTF Manager to be included in the FTF Stage command. If you do not specify this value, FTF uses the lqm value, specified in the FTF configuration file.
- **Source Queue Manager** – Name of the queue manager where the FTF Sender resides.
- **Config File** – The fully qualified path and filename for the FTF configuration file. This value is optional if the FTF\_CONFIG\_FILE environment variable is set. If you are using a queue to hold configuration values, do not enter a value in this field.
- **Config Queue** – Queue from which the configuration information is to be retrieved for this FTF instance on this node. The -cq argument points FTF to the queue name rather than to the standard configuration file.
- **Query** – Determines if the FTF Stage command queries the staging queue. If you specify this parameter, you cannot specify the Purge parameter.
- **Purge** – Determines if the FTF Stage command purges information from the staging queue that matches the specified File or FTFID options. If you specify this parameter, you cannot specify the Query parameter.

---

### **Note:**

If you specify a filename to be purged from the staging queue and multiple entries have the same filename, the file related to the first matching entry found is purged. If you need to specify a specific instance of a file in the staging queue, use that file's FTFID.

---

Using the 5250 User Interface  
FTF Stage

- **All** – Allows you to purge all files that are currently staged. If you use the All parameter, you must also use the Purge parameter. You cannot specify the All parameter with the Query parameter.

FTF Stage – Group 2

The following figure displays the second group of parameters used with the Stage FTF panel. Type the appropriate parameter values and press **[Page Up]** to see more parameters or press **[ENTER]** to execute the command.

Stage FTF (FTFSTAGE)

Type choices, press Enter.

Staged File . . . . .

FTF Transfer ID . . . . .

Seconds to wait for reply . . . . .

Display Version . . . . .

Character value

\*NO, \*YES

\*NO

Bottom

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display

F24=More keys

- You can enter the following parameters after you scroll down the first time:
- **Staged File** – The file that is purged from the staging queue. If you specify this argument, you cannot specify the Query or FTFID parameters.
  - **FTF Transfer ID** – FTFID of the file that is purged from the staging queue. If you specify this parameter, you cannot specify the Query or Staged File parameters.
  - **Seconds to wait for reply** – Time in seconds to wait for a reply to the query operation. This value does not apply to the purge operation.  
**Default value:** 300 seconds (5 minutes).

- **Display Version** – Returns the current FTF version, release, and patch number. Any command entered with this argument ignores all other arguments and returns the version information. **Valid Values:** YES, NO

## FTF Configuration

The Start FTF Config panel accommodates interfaces that process information in queues rather than files. This section describes the Start FTF Config parameters in groupings based on what appears when you press **[Page Down]** to scroll down.

### FTF Config – Group 1

The following figure displays the first group of parameters used with the Start FTF Config panel. Type the appropriate parameter values and press **[Page Down]** to see more parameters or **[ENTER]** to execute the command.

The screenshot shows a terminal window titled "Start FTF Config (FTFCFG)". The prompt "Type choices, press Enter." is displayed. Below it are several input fields with labels and dotted lines indicating where to type:

- Local Queue Manager . . . . .
- Config File . . . . .
- Options File . . . . .
- Node . . . . .
- Node File . . . . .

At the bottom right, there is a "More..." link. At the bottom left, there is a list of function key shortcuts:

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display  
F24=More keys

You can enter the following parameters on the Start FTF Config panel that appears before you scroll down:

## Using the 5250 User Interface

### *FTF Configuration*

- **Local Queue Manager** – Queue manager from which the FTFCFG command is issued. If this value is not specified, the FTFCFG command connects to the default queue manager that is set in the MQSeries configuration. Otherwise, whenever a command or interface starts up, it always tries to connect to the local queue manager (lqm). If no lqm value is specified, the command or interface attempts to connect to the specified default queue manager on platforms where MQSeries supports them.
- **Config File** – The fully qualified path and filename for the FTF configuration file. This value must be set on OS/390. In other operating systems, it is optional if the FTF\_CONFIG\_FILE environment variable is set.
- **Options File** – The fully qualified path and filename of a text file used to contain command-line arguments for the FTFCFG command. In the options file, you can set any of the command-line arguments that can be set for the FTFCFG command. Any values specified on the command line override the values in the options file.
- **Node** – The FTF node or nodes that are to be updated. This argument may be specified more than once or with multiple nodes separated by commas. If this argument is used with -nodefile, duplicates are eliminated.
- **Node File** – A file that contains a list of the FTF node names that need to be updated. If this argument is used with -node, duplicates are eliminated.

## FTF Config – Group 2

The following figure displays the second group of parameters used with the Start FTF Config panel. Type the appropriate parameter values and press **[Page Up]** to see more parameters or press **[ENTER]** to execute the command.



```
Start FTF Config (FTFCFG)

Type choices, press Enter.

Display Version . . . . . NO      *NO, *YES

                                                                 Bottom
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

You can enter the following parameter after you scroll down the first time:

- **Display Version** – Returns the current FTF version, release, and patch number. Any command entered with this argument ignores all other arguments and returns the version information.

## Starting an FTF Manager

The Start FTF Manager panel allows you to construct and execute a command that starts the specified FTF Manager. Type the appropriate parameter values and press **[Page Down]** for more parameters or press **[ENTER]** to execute the command.

**Using the 5250 User Interface**  
*Starting an FTF Manager*

Start FTF Manager (STRFTFMGR)

Type choices, press Enter.

Local Queue Manager . . . . .

Config File . . . . .

Config Queue . . . . .

Display Version . . . . .

Log File . . . . .

Node Name . . . . .

Job Name . . . . .

F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display  
F24=More keys

Character value

More...

You can enter the following arguments on the Start FTF Manager panel:

- **Local Queue Manager** – Queue manager from which the Start FTF Manager command is issued. If this value is not specified, the FTF Start Manager command connects to the default queue manager that is set in the MQSeries configuration. Otherwise, whenever a command or interface starts up, it always tries to connect to the local queue manager (lqm). If no lqm value is specified, the command or interface attempts to connect to the specified default queue manager on platforms where MQSeries supports them.
- **Config File** – The fully qualified path and filename for the FTF configuration file. This value is optional if the FTF\_CONFIG\_FILE environment variable is set. If you are using a queue to hold configuration values, do not enter a value in this field.
- **Config Queue** – Queue from which the configuration information is to be retrieved for this instance on this node. The cq argument points FTF to the queue name rather than to the standard configuration file.
- **Display Version** – Returns the current FTF version, release, and patch number. Any command entered with this argument ignores all other arguments and returns the version information.

212

*Tivoli Data Exchange User's Guide*

- **Log File** – File to which the FTF Manager writes a record of all data-transfer requests.
- **Node Name** – Associates the node name with the instance of FTF components.
- **Job Name** – The job name of the STRFTFMGR job to be submitted.

## Start FTF Manager – Group 2

The following figure displays the second group of parameters used with the Start FTF Manager panel. Type the appropriate parameter values and press **[Page Up]** to see more parameters or **[ENTER]** to execute the command.

The screenshot shows a terminal window titled "Start FTF Manager (STRFTFMGR)". The prompt "Type choices, press Enter." is displayed. Two parameters are shown: "Job Description" with a value of "FTFJOB0" and "Job Queue" with a value of "FTFV2". At the bottom, a list of function keys is provided: F3=Exit, F4=Prompt, F5=Refresh, F12=Cancel, F13=How to use this display, and F24=More keys. The word "Bottom" is also visible in the bottom right corner.

```
Start FTF Manager (STRFTFMGR)

Type choices, press Enter.

Job Description . . . . . 'FTFJOB0'
Job Queue . . . . . 'FTFV2'

F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys

Bottom
```

You can enter the following parameter after you scroll down the first time:

- **Job Description** – The job description of the STRFTFMGR job to be submitted.
- **Job Queue** – The job queue for the STRFTFMGR job to use.

# Starting an FTF Sender

The Start FTF Sender panel allows you to construct and execute a command that starts the specified FTF Sender. Type the appropriate parameter values and press [Page Down] to view more parameters or press [ENTER] to execute the command

Start FTF Sender (STRFTFSDR)

Type choices, press Enter.

Local Queue Manager . . . . .

Config File . . . . .

Config Queue . . . . .

Sync Queue Override . . . . .

Display Version . . . . .

Node Name . . . . .

Log File . . . . .

Character value  
\*NO, \*YES

F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display

F24=More keys

More...

You can enter the following arguments on the Start FTF Sender panel:

- Local Queue Manager** – The queue manager from which the FTF Start FTF Sender command is issued. If this value is not specified, the Start Sender command connects to the default queue manager that is set in the MQSeries configuration. Otherwise, whenever a command or interface starts up, it always tries to connect to the local queue manager (lqm). If no lqm value is specified, the command or interface attempts to connect to the specified default queue manager on platforms where MQSeries supports them.
- Config File** – The fully qualified path and filename for the FTF configuration file. This value must be set on OS/390. In other operating systems, it is optional if the FTF\_CONFIG\_FILE environment variable is set.

214

Tivoli Data Exchange User's Guide

- **Config Queue** – Queue from which the configuration information is to be retrieved for this FTF instance on this node. The -cq argument points FTF to the queue name rather than to the standard configuration file.
- **Sync Queue Override** – Determines which sync queue to use for processing as opposed to having FTF determine which one to use.
- **Display Version** – Returns the current FTF version, release, and patch number. Any command entered with this argument ignores all other arguments and returns the version information.
- **Node Name** – The FTF node or nodes that are to be updated. This argument may be specified more than once or with multiple nodes separated by commas.
- **Log File** – The file to which the FTF Sender writes a record of all data-transfer requests.

## Start FTF Sender – Group 2

The following figure displays the second group of parameters used with the Start FTF Sender panel. Type the appropriate parameter values and press **[Page Up]** or **[Page Down]** to view more parameters or press **[ENTER]** to execute the command

The screenshot shows a terminal window titled "Start FTF Sender (STRFTFSDR)". The prompt is "Type choices, press Enter." The screen displays several configuration options, each with a line for input and a default value in parentheses. The options are: "Options File", "Authorization Exit Enabled" (with sub-options \*NO and \*NO, \*YES), "Authorization Exit", "Authorization Entry", and "Portal Exit Enabled" (with sub-options \*NO and \*NO, \*YES). At the bottom, there is a "More..." prompt and a list of function key shortcuts: F3=Exit, F4=Prompt, F5=Refresh, F12=Cancel, F13=How to use this display, and F24=More keys.

```
Start FTF Sender (STRFTFSDR)

Type choices, press Enter.

Options File . . . . .

Authorization Exit Enabled . . . (*NO) (*NO, *YES)
Authorization Exit . . . . .

Authorization Entry . . . . .

Portal Exit Enabled . . . . . (*NO) (*NO, *YES)

More...
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

## Using the 5250 User Interface

### *Starting an FTF Sender*

You can enter the following parameter after you scroll down the first time:

- **Options File** – The fully qualified path and filename of a text file used to contain command-line arguments for the FTF Start Sender command. In the options file, you can set any of the command-line arguments that can be set for the FTF Start Sender command. Any values specified on the command line override the values in the options file.
- **Authorization Exit Enabled** – Activates the authorization module. The code for this module checks the user name, file, and access type contained in the FTFExitAuthInfo data structure against RACF via SAF. **Valid Values:** YES, NO
- **Authorization Exit** – Name of the service program that contains the authorization module.
- **Authorization Entry** – Name of the authorization function.
- **Connector Exit Enabled** – Activates the FTF Sender connector module. **Valid Values:** YES, NO

## Start FTF Sender – Group 3

The following figure displays the third group of parameters used with the Start FTF Sender panel. Type the appropriate parameter values and press **[Page Up]** to see more parameters or press **[ENTER]** to execute the command.

```

Start FTF Sender (STRFTFSDR)

Type choices, press Enter.

Connector Exit . . . . . █
Connector Entry . . . . . 
Connector Data . . . . . 

Job Name . . . . . 'FTFSDR' Character value
Job Description . . . . . 'FTFJOB' 
Job Queue . . . . . 'FTFV2' 

Bottom
F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys
  
```

You can enter the following parameters after you scroll down the second time:

- **Connector Exit** – Name of the service program that contains the connector exit.
- **Connector Entry** – Name of the function in the service program that contains the connector module.
- **Connector Data** – Data being passed through the FTF Sender to the connector. This value is optional.
- **Job Name** – Job name of the STRFTFSDR job to be submitted.
- **Job Description** – Job description of the STRFTFSDR job to be submitted.
- **Job Queue** – Job queue for the STRFTFSDR job to use.

## Starting an FTF Receiver

The Start FTF Receiver panel allows you to construct and execute a command that starts the specified FTF Receiver. Type the appropriate parameter values and press [ENTER] to execute the command.

**Using the 5250 User Interface**  
*Starting an FTF Receiver*

Start FTF Receiver (STRFTFRCV)

Type choices, press Enter.

Local Queue Manager . . . . .

Config File . . . . .

Config Queue . . . . .

Sync Queue Override . . . . .

Display Version . . . . .

Node Name . . . . .

Log File . . . . .

Character value

\*NO, \*YES

\*NO

More...

F3=Exit

F4=Prompt

F5=Refresh

F12=Cancel

F13=How to use this display

F24=More keys

You can enter the following arguments on the Start FTF Receiver panel:

- **Local Queue Manager** – Queue manager from which the FTF Start FTF Receiver command is issued. If this value is not specified, the Start FTF Receiver command connects to the default queue manager that is set in the MQSeries configuration. Otherwise, whenever a command or interface starts up, it always tries to connect to the local queue manager (lqm). If no lqm value is specified, the command or interface attempts to connect to the specified default queue manager on platforms where MQSeries supports them.
- **Config File** – The fully qualified path and filename for the FTF configuration file. This value must be set on OS/390. In other operating systems, it is optional if the FTF\_CONFIG\_FILE environment variable is set.
- **Config Queue** – Queue from which the configuration information is to be retrieved for this FTF instance on this node. The -cq argument points FTF to the queue name rather than to the standard configuration file.
- **Sync Queue Override** – Determines which sync queue to use for processing as opposed to having FTF determine which one to use.



- **Display Version** – Returns the current FTF version, release, and patch number. Any command entered with this argument ignores all other arguments and returns the version information. **Valid Values:** YES, NO
- **Node Name** – The FTF node or nodes that are to be updated. This argument may be specified more than once or with multiple nodes separated by commas.
- **Log File** – File to which the FTF Receiver writes a record of all data-transfer requests.

## Start FTF Receiver – Group 2

The following figure displays the second group of parameters used with the Start FTF Receiver panel. Type the appropriate parameter values and press **[Page Up]** to see more parameters or press **[ENTER]** to execute the command.

The screenshot shows a terminal window titled "Start FTF Receiver (STRFTFRCV)". The prompt is "Type choices, press Enter." Below this are several input fields and options:

- "Options File . . . . . " followed by a cursor.
- "Authorization Exit Enabled . . . " with "\*NO" selected and "\*NO, \*YES" as an option.
- "Authorization Exit . . . . . " followed by a cursor.
- "Authorization Entry . . . . . " followed by a cursor.
- "Connector Exit Enabled . . . . . " with "\*NO" selected and "\*NO, \*YES" as an option.
- A "More..." link at the bottom right.
- A footer with function key definitions: F3=Exit, F4=Prompt, F5=Refresh, F12=Cancel, F13=How to use this display, F24=More keys.

You can enter the following parameters after you scroll down the first time:

- **Options File** – The fully qualified path and filename of a text file used to contain command-line arguments for the FTF Start Receiver command. In the options file, you can set any of the command-line arguments that can be set for the FTF Start Receiver command. Any values specified on the command line override the values in the options file.

Using the 5250 User Interface  
Starting an FTF Receiver

- **Authorization Exit Enabled** – Activates the authorization module. The code for this module checks the user name, file, and access type contained in the FTFAuthInfo data structure against RACF via SAF. **Valid Values:** YES, NO
- **Authorization Exit** – Name of the function in the service program that contains the authorization module.
- **Authorization Entry** – Name of the authorization function.
- **Connector Exit Enabled** – Activates the FTF Receiver connector module. **Valid Values:** YES, NO

Start FTF Receiver – Group 3

The following figure displays the third group of parameters used with the Start FTF Receiver panel. Type the appropriate parameter values and press **[Page Up]** to see more parameters or **[ENTER]** to execute the command.

Start FTF Receiver (STRFTFRCV)

Type choices, press Enter.

Connector Exit . . . . .

Connector Entry . . . . .

Connector Data . . . . .

Job Name . . . . . 'FTFRCV' Character value

Job Description . . . . . 'FTFJOB0'

Job Queue . . . . . 'FTFV2'

Bottom

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display

F24=More keys

You can enter the following parameters after you scroll down the second time:

- **Connector Exit** – Name of the service program that contains the connector exit.

- **Connector Entry** – Name of the function in the service program that contains the connector module.
- **Connector Data** – Data being passed through the FTF Receiver to the connector. This value is optional.
- **Job Name** – Job name of the STRFTFRCV job to be submitted.
- **Job Description** – Job description of the STRFTFRCV job to be submitted.
- **Job Queue** – Job queue for the STRFTFRCV job to use.

## Starting the FTF Logger

The Start FTF Logger panel allows you to log information to two destinations, a local log file you specify when you start a FTF component and a log queue specified in the FTF configuration file. Using log queues allows you to collect log information without relying on a file. It can also allow for easier remote access to log information.

### FTF Logger – Group 1

Type the appropriate parameter values and press **[Page Down]** to see more parameters or press **[ENTER]** to execute the command.

## Using the 5250 User Interface

### Starting the FTF Logger

```
Start FTF Logger (STRFTFLOG)

Type choices, press Enter.

Local Queue Manager . . . . . █
Config File . . . . .
Config Queue . . . . .
Logger Queue . . . . .
Log Directory . . . . .

More...
F3=Exit  F4=Prompt  F5=Refresh  F12=Cancel  F13=How to use this display
F24=More keys
```

- **Local Queue Manager** – Local queue manager (lqm) from which the FTFLOG daemon is started. This value is required on OS/390 systems. On all other systems, if this value is not specified, the command attempts to connect to the specified default queue manager on platforms where MQSeries supports them.
- **Config File** – The fully qualified path and filename for the FTF configuration file. This value must be set on OS/390. In other operating systems, it is optional if the FTF\_CONFIG\_FILE environment variable is set.
- **Config Queue** – Queue from which the configuration information is to be retrieved for this FTF instance on this node. The cq argument points FTF to the queue name rather than to the standard configuration file.
- **Logger Queue** – Name of the log queue from which the log information is taken. You can specify only one log queue in any FTFLOG command.
- **Log Directory** – Name of the directory to which the log files are written.

## FTF Logger – Group 2

The following figure displays the second group of parameters used with the Start FTF Logger panel. Type the appropriate parameter values and press [Page Up] to see more parameters or [ENTER] to execute the command.

The screenshot shows a terminal window titled "Start FTF Logger (STRFTFLOG)". The prompt is "Type choices, press Enter." Below this are several input fields: "Log File", "Options File", "Display Version", "Job Name", "Job Description", and "Job Queue". The "Display Version" field has a dropdown menu with options "\*NO" and "\*NO, \*YES". The "Job Name" field has a dropdown menu with the value "'FTFLOG'". The "Job Description" field has a dropdown menu with the value "'FTFJOB0'". The "Job Queue" field has a dropdown menu with the value "'FTFV2'". At the bottom of the panel, there is a "Bottom" label and a list of function keys: "F3=Exit", "F4=Prompt", "F5=Refresh", "F12=Cancel", "F13=How to use this display", and "F24=More keys".

You can enter the following parameters after you scroll down the first time:

- **Log File** – Name of the log daemon's log file.
- **Options File** – The fully qualified path and filename of a text file used to contain command-line arguments for the FTF Start Log command. In the options file, you can set any of the command-line arguments that can be set for the FTF Start Log command. Any values specified on the command line override the values in the options file.
- **Display Version** – Returns the current FTF version, release, and patch number. Any command entered with this argument ignores all other arguments and returns the version information. **Valid Values:** YES, NO
- **Job Name** – The job name of the STRFTFLOG job to be submitted.
- **Job Description** – The job description of the STRFTFLOG job to be submitted.

## Using the 5250 User Interface

### *Logging off FTF*

- **Job Queue** – The job queue for the STRFTFLOG job to use.

## Logging off FTF

When you have returned to the FTF menu after performing your processing, you can press **[F3]** to get to the main menu. Press **[F3]** again to sign off of AS/400 so that you may perform other processes.







# Using the FTF GUI

Tivoli Data Exchange (TDE) includes a graphical user interface (GUI) written using the Java language. This GUI, referred to in this chapter as the TDE GUI, allows you to interactively submit and monitor data-transfer requests in the Windows NT and UNIX environments. It allows you to interactively perform all the tasks that the TDE commands and APIs perform.

The TDE GUI is useful for administrators monitoring data-transfer requests and submitting ad hoc requests. It is also useful for developers who want to work out the details of a particular data-transfer request before writing code or scripts to perform the transfer. This section describes how to access and use the TDE GUI.

It includes the following information:

<b>Section</b>	<b>Title</b>
Starting the Tivoli Data Exchange GUI	228
Viewing Data-Transfer Status Information	229
Monitoring Status Information	237
Submitting a Data-Transfer Request	238
Pinging Tivoli Data Exchange Components	251
Staging a Data-Transfer Request	252
Canceling a Transfer Request	255
Shutting Down Tivoli Data Exchange Components	257
Stopping Monitoring	258

## Assumptions

The information in this manual assumes that:

- The TDE GUI has been successfully installed on a machine that has access to the files being transferred and to all related TDE components.
- If you are running the TDE GUI on a UNIX platform, this chapter assumes the configuration steps listed in “Installing TDE on UNIX: Configuring the TDE GUI” in the *Tivoli Data Exchange Installation Guide* have been successfully completed.
- Users will be given appropriate administrative privileges for the functionality in the TDE GUI.

## Starting the Tivoli Data Exchange GUI

This section contains information about starting the TDE GUI in all supported operating systems. The following operating systems are currently supported:

- Win 32
- UNIX (Solaris, HP-UX, and AIX)

To start the TDE GUI, use one of the following methods:

- (Win 32 only) From the Win 32 **Start** menu on your task bar, select **Programs>TDE > TDE Status GUI**.

**- Or -**

- (All platforms) At a command prompt, enter the following command

```
ftfgui.cmd qMgrName
```

Where *qMgrName* is the name of the queue manager to connect to.

When you perform either of these steps, the TDE GUI starts, allowing you to specify filter criteria for viewing status information.

## Viewing Data-Transfer Status Information

The TDE GUI allows you to view data-transfer status information according to filter information that you specify. After this status information is displayed, you can view detailed status information and delete information from the status and stage queues.

### Filtering Status Information

1. Filtering data-transfer status information allows you to reduce the status record display so it includes only records that correspond to criteria you set. For instance, you can filter status information to include only status records logged within a specified time period.

---

#### Note:

When you use the FTF Status Filter, the start time and end time parameters are expressed in military time format and you cannot use asterisks (\*) with time/date fragments.

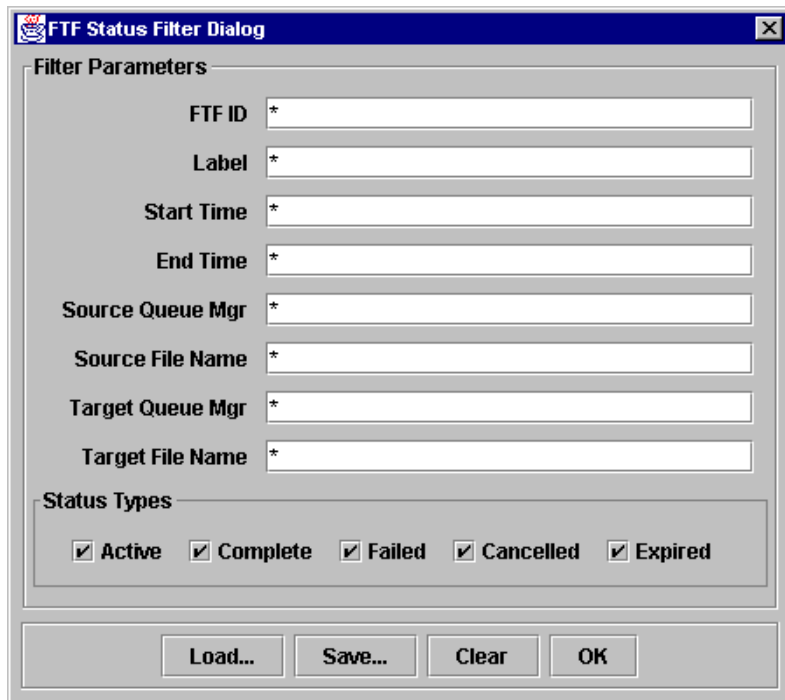
---

After you have set filter criteria, you can save them and reuse them at a future time.

To filter status information, you must use the *FTF Status Filter Dialog* window. Although this window appears whenever you start the TDE GUI, you can also gain access by selecting **Options>Set Filter Parameters...**

## Using the FTF GUI

### Viewing Data-Transfer Status Information



To filter data-transfer status information, follow these steps:

1. If the GUI is monitoring status information for changes, select **Actions>Stop** to stop monitoring.

---

### Note:

You cannot filter or query data-transfer status information while the TDE GUI is monitoring.

---

2. From the **Options** menu, select **Set Filter Parameters** to access the *FTF Status Filter Dialog* window.
3. If you want to load an existing filter, click the **Load...** button. Select the filter file from the file dialog box that appears.

- Or -

If you want to clear the contents of the *FTF Status Filter Dialog* window, click the **Clear** button.

4. Enter or modify the filter information, as appropriate. Data-transfer status information must satisfy all filter criteria to appear as filtered status information. The asterisks that appear in the fields return records with all possible values in that specific field. If you leave a field blank, it also returns all records with all possible values for that field.
  - **FTF ID** – Determines the FTF ID for which status information is returned.
  - **Label** – Determines the label for which status information is returned. Labels are user-defined identifiers created for grouping data-transfer requests.
  - **Start Time** – Contains the beginning of the date range for which status information is returned. If you specify the beginning of the date range, but no end, TDE returns all status information logged since the specified date. **Format:** YYYYMMDDhhmmss, where YYYY is the four-digit year, MM is the two-digit month, DD is the day, hh is the hour (24-hour format), mm is the minute, and ss is the second.
  - **End Time** – Contains the end of the date range for which status information is returned. If you specify the end of the date range, but no beginning, TDE returns all status information logged until the specified date. **Format:** YYYYMMDDhhmmss, where YYYY is the four-digit year, MM is the two-digit month, DD is the day, hh is the hour (24-hour format), mm is the minute, and ss is the second.

## **Working with Date Values**

The TDE GUI allows you to specify a partial date value. To specify a partial date, you must specify everything to the left of the smallest specified value. In other words, if you want to specify an hour, you must also specify the day, the month and the year.

- **Source Queue Mgr** – Determines the TDE Sender for which status information is returned.

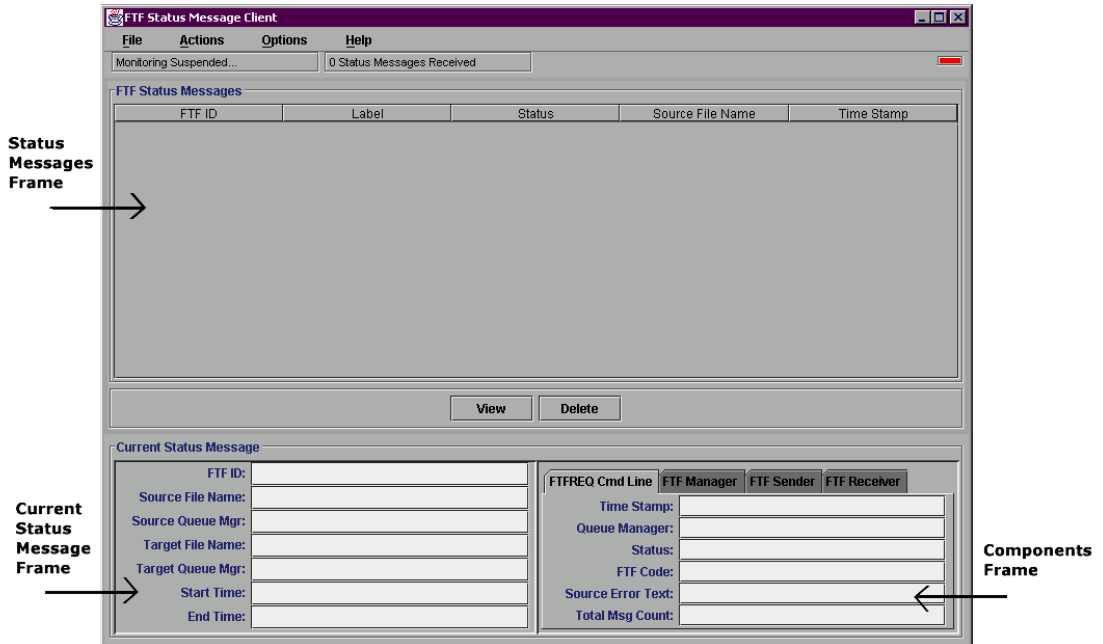
## Using the FTF GUI

### *Viewing Data-Transfer Status Information*

- **Source File Name**– Determines the fully qualified path and file name of the source file(s) for which status information is returned. You can expand the filter information by using wildcards in the source file designation.
  - **Target Queue Mgr** – Determines the TDE Receiver for which status information is returned.
  - **Target File Name**– Determines the fully qualified path and file name of the target file(s) for which status information is returned. You can expand the filter information by using wildcards in the target file designation.
5. Select the status types for which message information is included. To select a status, check its check box. To deselect a status, uncheck its check box. You can select any of the following status types:
    - Active
    - Complete
    - Failed
    - Canceled
    - Expired
  6. If you want to save the filter criteria, click the **Save...** button and enter the name and path of the file you want to contain the filter.
  7. Click **OK**.
  8. From the **Actions** menu, select **Start** to view status information.

## Viewing Status Information

The main window of the TDE GUI (the *FTF Status Message Client* window) contains three frames: the Status Messages frame, the Current Status Message frame, and the Component frame. The following diagram displays the *FTF Status Message Client* window.



#### **The Status Messages Frame**

The Status Messages Frame displays details of each data-transfer status record. The following information is displayed for each status message:

- FTF ID
- Label
- Status
- Source File Name
- Time Stamp

#### **Altering Status Record Display**

To sort status records by a specific value, click the header for that value. For instance, click the **FTF ID** header to sort by the FTFID.

To change the length of a field's display area, move your mouse pointer over the field header's right boundary and drag the boundary to the desired width.

To change the order the columns appear, click a column's header and drag it to the location you want the column to appear.

#### **Current Status Message Frame**

The Current Status Message Frame lists the following information for the data-transfer request related to the status record highlighted in the Status Messages frame:

- FTF ID
- Source file name
- Source Queue Manager
- Target file name
- Target Queue Manager
- Start time
- End time

#### **Components Frame**

The Components Frame lists data-transfer request information at four different spots in the data-transfer process: the command line from which the data-transfer request was made, the TDE Manager used in the data-transfer request, the TDE Sender used to send the data being transferred, and the TDE Receiver used to receive the data.



The following information is listed for each component:

- **Time Stamp** – Indicates the time and date at which the data transfer reached the current point in data-transfer process.
- **Queue Manager** – Indicates the queue manager on which that part of the data-transfer process took place.
- **Status** – Indicates the data transfer's status at the specified queue manager.
- **FTF Code** – Indicates the TDE status code associated with the data-transfer process at the specified queue manager.
- **Source Error Text** – Lists descriptive text associated with the TDE code value.
- **Total Msg Count** – Number of data messages used to transfer the data.

## Viewing Detailed Status Information

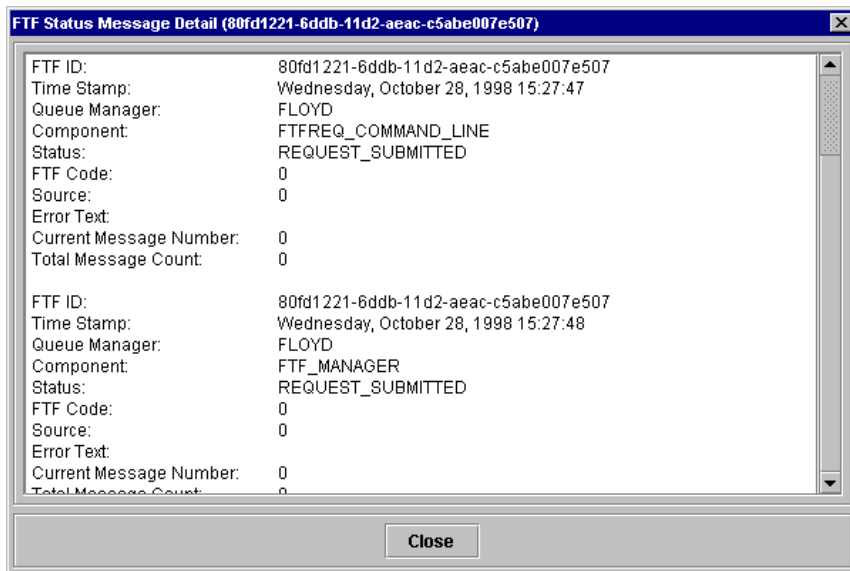
To view detailed status information for a specific status record, perform one of these steps:

- Double-click the record.
- With the record highlighted, click the **View** button.

The *FTF Status Message Detail* window appears.

## Using the FTF GUI

### *Viewing Data-Transfer Status Information*



The following information appears for each point in the data-transfer process:

- FTFID
- Time Stamp
- Queue Manager
- Component
- Status
- FTF Code
- Source
- Error Text
- Current Message Number
- Total Message Count

When you finish viewing the detailed status information, click the **Close** button.

## Deleting Status Records

Deleting status records removes them from the status queue. You cannot use TDE to restore a deleted status record and no one else can view that status record.

To delete a status record, follow these steps:

1. In the FTF Status Messages frame, click the status record you want to delete.
2. Click the **Delete** button. The record is deleted from the status queue.

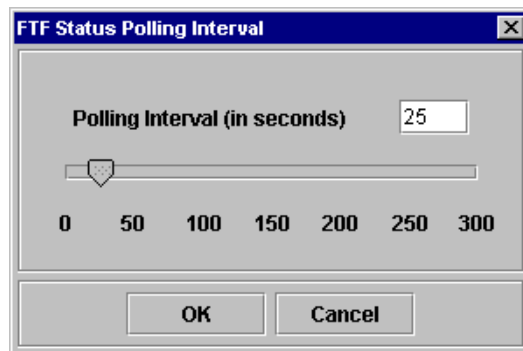
## Monitoring Status Information

After you have displayed status information according to filter criteria, you can use the TDE GUI to monitor the data-transfer requests that conform to the filter criteria. The information displayed in the status records is updated once per polling interval. The polling interval is a value you can set that determines how often the status information is updated.

To monitor status information, select **Actions>Start**. The red indicator in the upper right corner of the *FTF Status Message Client* window turns green to show that monitoring is occurring.

## Changing the Polling Interval

To change the polling interval—the time period between status information updates—select **Options>Set Polling Interval...**. The *FTF Status Polling Interval* window appears.



To specify an interval value, type the value in the *Polling Interval (in seconds)* field or move the slider to the desired value. Click **OK** to apply the change or **Cancel** to disregard the change.

## Submitting a Data-Transfer Request

The TDE GUI allows you to submit data-transfer requests based on parameters that you set within the TDE GUI.

To submit a data-transfer request, follow these steps:

1. Select **Options> FTF Request Dialog...** The *FTF Request Dialog* appears:

FTF Request Dialog

Local Queue Manager: PROD11A

Originating Queue Manager: PROD11A

Source Queue Manager: PROD11A

Target Queue Manager: PROD11A

Source File Information

... Options

Target File Information

... Options

Stage Action

FTFD From Stage:

Options

Job Options User Options Wait For Reply:

Configuration File

... C:\FTF\ftfconfig.ini

Reply Wait Time: 30

Exit Information

Submit Done

2. Enter the appropriate values in the following fields:
  - **Originating Queue Manager** – Determines the queue manager on which the data transfer's TDE Manager operates. You must enter an Originating Queue Manager value.

- **Source Queue Manager** – Determines the queue manager on which the data transfer's TDE Sender operates. You must enter a Source Queue Manager value.
  - **Target Queue Manager** – Determines the queue manager on which the data transfer's TDE Receiver operates. You must enter a target queue manager value.
3. Enter the fully qualified path and filename for the source file in the **Source File Information** field. To browse for a source file value, click the ... button.
  4. To set options associated with the source file, click the **Options** button. For more information about entering source file options, see “Setting Source File Options” on page 241.
  5. Enter the fully qualified path and filename for the target file in the **Target File Information** field. To browse for a target file value, click the ... button.
  6. To set options associated with the target file, click the **Options** button. For information about setting target file options, see “Setting Target File Options” on page 243.
  7. To send a file from a staging queue, specify its **FTFID** or its name in the **Source File Information** and check the **From Stage:** check box.
  8. To set options that govern how the job runs, click the **Job Options** button. For more information about setting job options, see “Setting Job Options” on page 245.
  9. To set user-specified options, click the **User Options** button. For more information about setting user options, see “Setting User Options” on page 247.
  10. To wait until you receive a message from the Originating Queue Manager indicating success or failure before this instance of the GUI can be used again, check the **Wait for Reply** check box. (If no reply is received, the GUI can be used again, but the data transfer continues to be processed.)
  11. To specify the configuration file that governs the TDE data transfer, enter the appropriate value in the **Configuration File** field. To browse for the configuration file, click the ... button.
  12. To specify a time period to wait for a reply from the TDE Manager, enter the appropriate value in the **Reply Wait Time** field. This value is measured in seconds.

## Using the FTF GUI

### *Submitting a Data-Transfer Request*

13. To set user exit information, click the **Exit Information** button. For more information about setting user exits, see “Setting User Exits” on page 248.
14. To submit the data-transfer request, click the **Submit** button. To exit the window without submitting a data-transfer request, click the **Done** button.

---

#### **Note:**

When a data-transfer request fails and the file mode for the TDE Receiver is set to “Create” or “Replace”, the TDE Receiver deletes the file. If the transfer request fails before the receiver starts processing to the target file, cleanup is not required. However, if the mode is set to “Create” or “Replace”, and the TDE Receiver has started writing data to the target file and the data-transfer request fails, the TDE Receiver deletes the file.

---

## Setting Source File Options

Source file options allow you to determine characteristics of the file being transferred. To set source file options, follow these steps:

1. In the *FTF Request Dialog*, click the **Options** button next to the **Source File Information** field. The *Source File Options* window appears:



The screenshot shows a window titled "Source File Options" with a standard Windows-style title bar (minimize, maximize, close buttons). The window contains several sections with labels on the left and options on the right:

- Source File Name:** A text input field.
- Stage Options:** Three checkboxes: ☐ Staged, ☐ Stage Only, and ☐ Stage Persistent.
- Data Options:** Two checkboxes: ☐ Data Persistent and ☐ Compress.
- File Type:** Two radio buttons: ☐ Binary File and ☐ Text File.

At the bottom right of the window is a "Done" button. The window has a scroll bar at the bottom.

2. Set the following values, as appropriate.
  - Check the **Staged** check box to store the messages that compose the data being transferred in a staging queue until the data transfer is complete. Staging allows the file to be resent from the staging queue, rather than from the original file. Staging is appropriate if the file does not permanently exist on the machine from which the TDE Sender is retrieving data.

## Using the FTF GUI

### *Submitting a Data-Transfer Request*

- Check the **Stage Only** check box to place the data messages on the staging queue, but not send them to a destination.
- Check the **Stage Persistent** check box to keep staging area data if MQSeries is stopped or recycled.
- Check the **Data Persistent** check box to make the files being transferred persistent. Persistent files still exist after an MQSeries recycle. If you select this option, you increase TDE's recovery ability, but decrease its performance.
- Check the **Compress** check box to invoke the compression algorithm included with TDE.
- Select whether the file being transferred is a **Binary File** or a **Text File**.

The following fields are optional:

- Enter the type of file to be transferred in **Source File Type**. This value must be the same as one of the FileTypes values specified in the TDE configuration file.
- Enter the table name, record format name, whether the data is fixed length, and delimiter character in the **Source File Type Data** field. Source File Type Data takes the following form:

`"TBL=tableName;RCDFMT=recordFmtName;FIXLEN=isFixed;  
DLM=delimiterChar"`

Where:

- **TBL** *tableName* – Contains the name of the physical or logical file located in `lib/filename(mbrname)`. **Default value:** If *mbrname* is not specified, the first member name is used.
- **RCDFMT** *recordFormatName* – Determines the source file's record format. **Default value:** If this value is not specified, the first record format is used.
- **FIXLEN** *fixedLength* – Specifies whether the record will have a fixed length at the target file. **Valid values:** Y, N **Default value:** N
- **DLM** *delimiterChar* – Contains the character to be used as the delimiter. **Default value:** If this value is not specified, a comma is used.



3. Click the **Done** button to set the source file options and return to the *FTF Request Dialog* window.

## Setting Target File Options

Target file options allow you to determine characteristics of the transferred file at its destination. To set target file options, follow these steps:

1. In the *FTF Request Dialog*, click the **Options** button next to the **Target File Information** field. The *Target File Options* window appears:

The screenshot shows the 'Target File Options' dialog box with the following fields and options:

- Target File Name:** [Text input field]
- File Type:** Radio buttons for ☐ Binary File and ☐ Text File
- Target File Type:** [Text input field]
- Target File Type Data:** [Text input field]
- File Mode:** Radio buttons for ☐ Create File, ☐ Append File, and ☐ No Replace File
- Options:** ☐ Create Directory
- MVS Options:**
  - File Organization:** Radio buttons for ☐ Partitioned File and ☐ Sequential File
  - Directory Block Count:** [Text input field with value 0]
  - Record Format:** Radio buttons for ☐ Fixed, ☐ Variable, ☐ Fixed Block, and ☐ Variable Block
- Logical Records:** [Text input field with value 0]
- Unit Name:** [Text input field]
- Block Size:** [Text input field with value 0]
- Volsr:** [Text input field]
- Allocation Format:** Radio buttons for ☐ Cylinder, ☐ Block, and ☐ Track
- Primary Allocation Size:** [Text input field with value 0]
- Secondary Allocation Size:** [Text input field with value 0]
- Text Wrap:** Radio buttons for ☐ Wrap, ☐ Fail, and ☐ Truncate
- Done** button

2. Set the following values, as appropriate:
  - Select whether the file being transferred is a **Binary File** or a **Text File**. The value set for the target file type must agree with the value of the source file type.

The following fields are optional:

## Using the FTF GUI

### Submitting a Data-Transfer Request

- Enter the appropriate **Target File Type**. This value must be the same as one of the FileTypes values specified in the TDE configuration file.
- Enter the table name, the record format name, the delimiter character, and if the data is of a fixed length in **Target File Type Data**. **Target File Type Data** takes the following form:

“TBL=*tableName*;RCDFMT=*recordFormatName*;FIXLEN=*N*;  
DLM=*delimiterChar*”

Where:

- **TBL** *tableName* – Contains the name of the physical or logical file located in lib/filename(*mbrname*). **Default value:** If *mbrname* is not specified, the first member name is used.
  - **RCDFMT** *recordFormatName* – Determines the target file’s record format. **Default value:** If this value is not specified, the first record format is used.
  - **FIXLEN** *fixedLength* – Specifies whether the record will have a fixed length at the target file. **Valid values:** Y, N **Default value:** N
  - **DLM** *delimiterChar* – Contains the character to be used as the delimiter. **Default value:** If this value is not specified, a comma is used.
3. Select the appropriate file mode. The **Create** option creates the target file if it does not already exist. If the target file does exist, it is overwritten. The **Append File** option creates the target file if it does not already exist. If the target file does exist, the transferred data is appended. The **No Replace File** option overwrites the target file if it already exists.
  - Check the **Create Directory** check box to create any directory structures required for the target file.
  3. If the data is being transferred to an OS/390 environment, set the following options, as appropriate:
    - Select the appropriate file organization: **Partitioned File** or **Sequential File**.
    - Enter the appropriate block count value.

- Select the appropriate record format: **Fixed**, **Variable**, **Fixed Block**, or **Variable Block**.
- Enter the appropriate number of logical records in the **Logical Records** field.
- Enter the appropriate block size in the **Block Size** field.
- Enter the target file's unit name in the **Unit Name** field.
- Enter the target file's volume in the **Volser** field.

### **Specifying an Esoteric Unit Name**

To specify an esoteric name for the OS/390 UNIT value, follow these steps:

- Do not set a value in the MVSVOLUME stanza of the TDE configuration file.
- Specify the unit value in the *Unit Name* field or in the MVSUNITNAME stanza in the TDE configuration file on either the TDE Sender or the TDE Receiver.

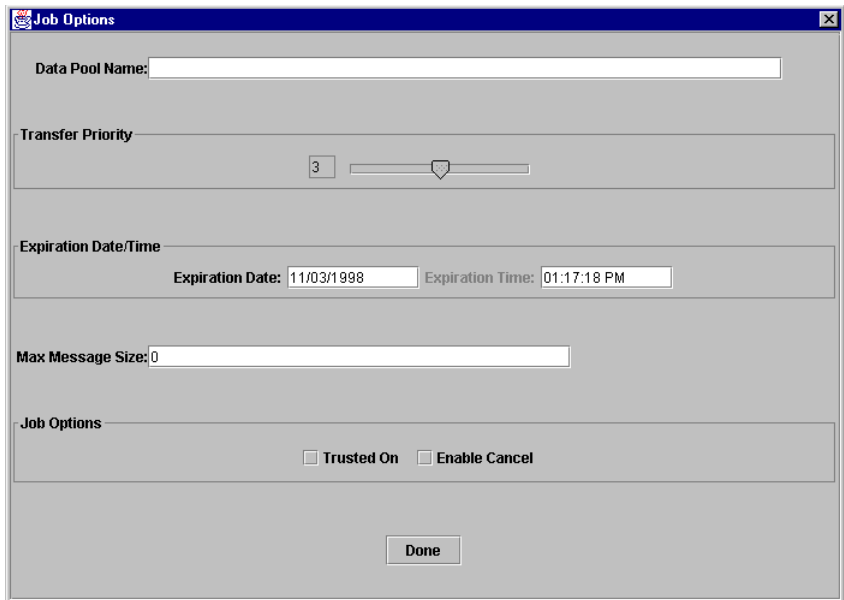
- Enter the appropriate allocation format: **Cylinder**, **Block**, or **Track**.
  - Enter the primary allocation size in the **Primary Allocation Size** field.
  - Enter the secondary allocation size in the **Secondary Allocation Size** field.
  - Enter the appropriate text wrap value: **Wrap**, **Fail**, or **Truncate**.
4. Click the **Done** button to set the target file options and return to the *FTF Request Dialog* window.

## **Setting Job Options**

Job options set conditions under which the data-transfer request runs. To set job options, click the **Job Options** button on the *FTF Request Dialog* window. The *Job Options* window appears.

## Using the FTF GUI

### Submitting a Data-Transfer Request



The screenshot shows a dialog box titled "Job Options" with a standard Windows-style title bar (minimize, maximize, close buttons). The dialog contains several input fields and checkboxes:

- Data Pool Name:** A text input field.
- Transfer Priority:** A slider control with a numeric box on the left showing the value "3".
- Expiration Date/Time:** Two text input fields. The first is labeled "Expiration Date:" and contains "11/03/1998". The second is labeled "Expiration Time:" and contains "01:17:18 PM".
- Max Message Size:** A text input field containing the value "0".
- Job Options:** A section containing two checkboxes: "Trusted On" and "Enable Cancel", both of which are currently unchecked.
- Done:** A button at the bottom center of the dialog.

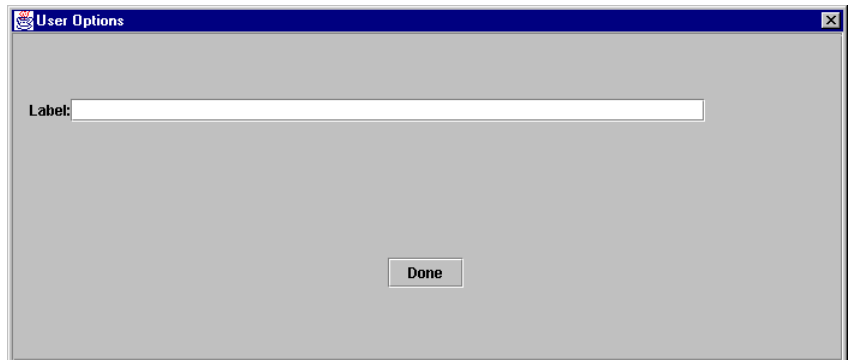
To set job options, follow these steps:

1. Enter the appropriate values in the following fields:
  - **Data Pool Name** – Determines the name of the data pool to be used for the data transfer. If you do not enter a value, TDE uses the default value specified in the TDE configuration file.
  - **Transfer Priority** – Determines the data transfer's priority. **Valid values:** 1 (highest) - 5 (lowest) **Default value:** 3.
  - **Expiration Date** – Determines the expiration date for the data-transfer request. **Format:** MM/DD/YYYY, where MM is the numeric month, DD is the numeric day, and YYYY is the four-digit year.
  - **Expiration Time** – Determines the expiration time for the data-transfer request. **Format:** hh:mm:ss AM/PM, where hh is the hour (twelve-hour clock), mm is the minute, ss is the second, and AM/PM is AM or PM.
  - **Max Message Size** – Sets a message size value that overrides the MQSeries message size value. This value is set in KB. **Valid values:** 1-3906 KB

2. If you want the data transfer to be trusted, check the **Trusted On** check box. A trusted data transfer sacrifices file-recovery ability to allow for a faster transfer. This option should be used only when file recovery is assured.
3. If you want to be able to cancel the data transfer after it is submitted, check the **Enable Cancel** check box.
4. Click the **Done** button to set the job options and return to the *FTF Request Dialog* window.

## Setting User Options

User options allow you to set user-specified options concerning the data transfer you are requesting. To set user options, click the **User Options** button on the *FTF Request Dialog* window. The *User Options* window appears.



To set user options, follow these steps:

1. Insert the data transfer's label value in the **Label** field. Labels are user-specified qualifiers you can use to create data-transfer groupings for viewing file status information. Labels can be up to 20 characters in length.
2. Click the **Done** button to set the user options and return to the *FTF Request Dialog* window.

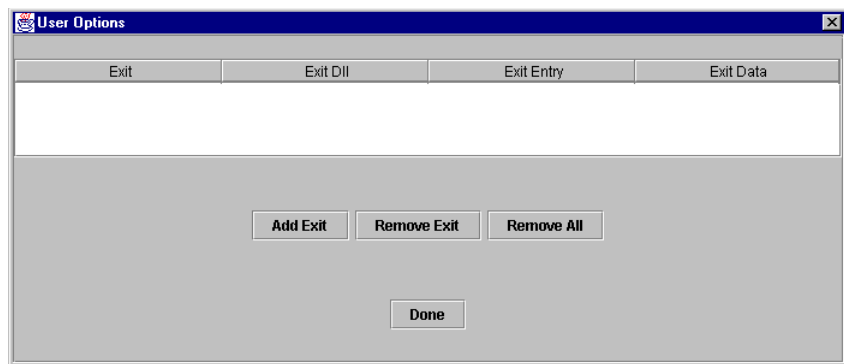
## Using the FTF GUI

### *Submitting a Data-Transfer Request*

## Setting User Exits

User exits allow you to plug your own modules into TDE to execute custom business logic. For more information about creating user exits, see Chapter 5, “Tivoli Data Exchange User Exits”.


To set user exits, click the **Exit Information** button on the *FTF Request Dialog* window. A window appears that allows you to specify the user exits that will be invoked when the data-transfer request is processed. This window allows you to add a user exit, remove a user exit, and remove all user exits.



### Adding a User Exit

To add a user exit, follow these steps:

1. On the *User Options* window, click the **Add Exit** button. The *Exit Entry Information* window appears:



The dialog box titled "Exit Entry Information" contains four input fields: "Exit:" with the value "3", "Exit Dtl:", "Exit Entry:", and "Exit Data:". At the bottom are "Ok" and "Cancel" buttons.

2. Enter the appropriate values in the following fields:

- **Exit Number** – Determines the number of the exit you want to invoke. The following table lists the exits and their numbers.

TDE Exit No.	Executing Node	TDE Component	Description
Exit 3	Originating Queue Manager	TDE Manager	Manager pre-process exit. If specified, this is the first exit executed in the data-transfer request.
Exit 4	Originating Queue Manager	TDEManager	Manager post-process. If specified, this is the last exit executed in the data-transfer request.  <b>Note:</b> This exit's execution does not depend on the success or failure of the data-transfer request.
Exit 5	Source Queue Manager	TDE Sender	Sender pre-process exit. This exit is invoked before the TDE Sender reads the source file's contents.
Exit 6	Source Queue Manager	TDESender	Sender post-process exit. The TDE Sender invokes this exit after the source file's contents have been read and processed for transfer to the target node.  <b>Note:</b> This exit is invoked regardless of whether or not the TDE Sender is able to successfully process the source file.

## Using the FTF GUI

### Submitting a Data-Transfer Request

TDE Exit No.	Executing Node	TDE Component	Description
Exit 7	Destination Queue Manager	TDEReceiver	Receiver pre-process exit. The TDE Receiver invokes this exit before it begins to write the target file.
Exit 8	Destination Queue Manager	TDE Receiver	Receiver post-process exit. The TDE Receiver invokes this exit after the target file has been processed and all the source data has been written to it.  <b>Note:</b> The receiver post-process exit is invoked by the TDE Receiver regardless of whether the target file is processed successfully.
Exit 9	Source Queue Manager	TDE Sender	Invokes the sender connector exit.
Exit 10	Source Queue Manager	TDE Receiver	Invokes the receiver connector exit.

---

### Note:

- If you are running connectors on a Solaris 2.5.1 operating system, you must install Solaris Patch 103627.
- 

- **Exit DLL** – Name of the DLL, shared object, load module, or service program that contains the exit module.
  - **Exit Entry** – The name of the function in the DLL to be invoked. This value is case sensitive.
  - **Exit Data** – An optional command-line argument where user-specified data may be passed to the exit. If you use the sample exit that comes with TDE, this argument contains the commands to be executed by the exit module.
3. Click the **Done** button to set the user options and return to the *FTF Request Dialog* window.



## Removing User Exits

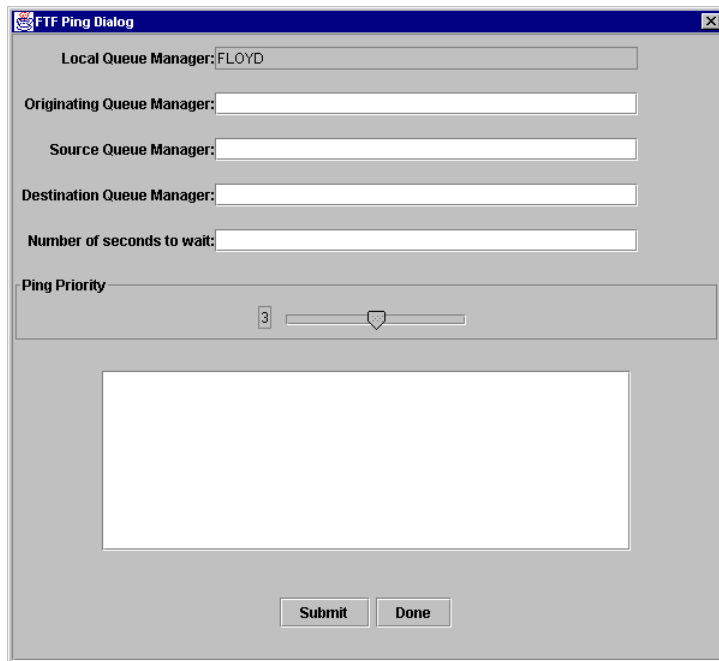
To remove a single user exit, select the user exit you want to remove, then click the **Remove Exit** button.

To remove all specified user exits, click the **Remove All** button.

## Pinging Tivoli Data Exchange Components

To ping TDE components, follow these steps:

1. Select **Options>FTF Ping Dialog...**. The *FTF Ping Dialog* window appears:



The screenshot shows the 'FTF Ping Dialog' window. It has a title bar with the text 'FTF Ping Dialog'. The main area contains the following fields and controls:

- Local Queue Manager:** A text box containing the value 'FLOYD'.
- Originating Queue Manager:** An empty text box.
- Source Queue Manager:** An empty text box.
- Destination Queue Manager:** An empty text box.
- Number of seconds to wait:** An empty text box.
- Ping Priority:** A section containing a numeric input field with the value '3' and a horizontal slider control.
- Buttons:** At the bottom right, there are two buttons labeled 'Submit' and 'Done'.

2. Enter the appropriate values in the following fields:

## Using the FTF GUI

### *Staging a Data-Transfer Request*

- **Originating Queue Manager** – Determines the TDE Manager to be included in the ping. If you do not specify this value, TDE uses the Local Queue Manager value specified above.
  - **Source Queue Manager** – Determines the TDE Sender to be included in the ping.
  - **Destination Queue Manager** – Determines the TDE Receiver to be included in the ping.
  - **Number of seconds to wait** – Determines the amount of time, in seconds, to allow for processing the ping request. **Valid values:** 1-32767
  - **Ping Priority** – Determines the ping's priority. **Valid values:** 1 (highest) - 5 (lowest) **Default value:** 3.
3. To submit the ping request, click the **Submit** button. The status information produced from the ping operation is displayed in the data window at the bottom of the window. To exit the window without submitting a ping request, click the **Done** button.

## Staging a Data-Transfer Request

Staging the data-transfer request allows you to send the file from the staging queue rather than from the original file. Staging is appropriate if the file being transferred does not permanently exist on the TDE Sender. You can also use the FTF Stage Dialog window to query the staging queue for a list of its contents, purge all currently staged data-transfer requests, or purge specific staged data transfer requests.

To stage a data-transfer request, follow these steps:

1. Select **Options>FTF Stage Dialog....** The *FTF Stage Dialog* window appears.

FTF Stage Dialog

Local Queue Manager: PROD11A

Originating Queue Manager: PROD11A

Source Queue Manager: PROD11A

Purge FileName:

Purge FTId:

Stage Action

☒ Query ☐ Purge ☐ PurgeAll

Options

☒ Wait For Reply: 30

Configuration File

... C:\FTF\ftfconfig.ini

Submit Done

2. Set the following values as appropriate.
  - **Originating Queue Manager** – Determines the TDE Manager to be included in the staged data-transfer request. If you do not specify this value, TDE uses the Local Queue Manager value specified above.
  - **Source Queue Manager** – Determines the TDE Sender component to be included in the staged data-transfer request.

## Using the FTF GUI

### *Staging a Data-Transfer Request*

- **Purge File Name** – Determines the file that is purged from the staging queue. If you specify the name of the file, TDE purges the first file that matches the name.
  - **Purge FTFid** – Determines the FTFID of the file that is purged from the staging queue.
3. Select one of the Stage Action options, as appropriate:
- **Query** – Determines that the FTFSTAGE command queries the staging queue.
  - **Purge** – Determines that the FTFSTAGE command will purge the file specified in the **Purge File Name** or **Purge FTFid** fields.
  - **PurgeAll** – Purges all files that are currently staged.

Check the following Options box as appropriate:

- **Wait for Reply** – Determines the time to wait for a reply to the query operation. This value does not apply to the purge operation. Specified in seconds. **Default value:** 30 seconds
4. To select a configuration file other than the default, click the ... button in the **Configuration File** section and select the desired file.
5. Click the **Submit** button to process the stage request. Click the **Done** button to exit the window without processing the stage request.

## Canceling a Transfer Request

The TDE GUI allows you to cancel an active data-transfer request. When TDE processes a cancellation request, the transaction is immediately stopped and purged. If you try to cancel a transaction that has already been completed, the cancellation request is ignored.

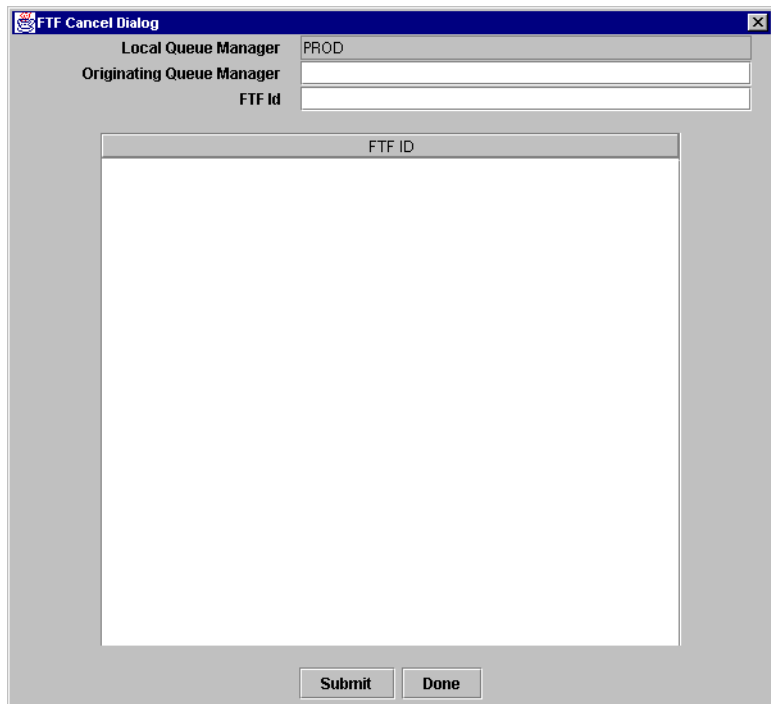
---

### Note:

For the cancel request to function properly, TDE must be actively monitoring and the Manager must already have received the request.

---

1. Select **Options>FTF Cancel Dialog....** The *FTF Cancel Dialog* window appears.



## Using the FTF GUI

### *Canceling a Transfer Request*

The FTFIDs appear in the **FTF ID** data window for all data-transfer requests for which no TDE Receiver is available.

2. Enter the FTFID of a data-transfer request that you wish to cancel and its associated originating queue manager.

**-Or-**

Select the desired FT ID in the **FTF ID** data window. If you selected an FTF ID value, its values populate the following fields:

- **Originating Queue Manager**
- **FTF Id**

---

### **Note:**

When the TDE Receiver component becomes available, all FTF ID values requiring that component are removed from the **FTF ID** data window.

---

3. Click the **Submit** button to process the cancellation request. Click the **Done** button to exit the window without processing the cancellation request.

# Shutting Down Tivoli Data Exchange Components

The TDE GUI allows you to shut down TDE components running on a specified node.

The TDE components being shut down do not process the shutdown message while they are processing a data-transfer request. For example, if the TDE Sender is transferring a large file, it does not read the shutdown message from the control queue until it finishes processing its data. Ending the TDE Sender by other means may cause the failure of the data-transfer request.

To shut down components, follow these steps:

1. Select **Options>FTF End Dialog....** The *FTF End Dialog* window appears.

FTF End Dialog

Local Queue Manager: PROD11A

Components Host Node: PROD11A

☐ Terminate With Pending Work:

Number of seconds to wait: 0

Components To End

☒ Manager: ☒ Sender: ☒ Receiver: ☒ Logger:

Submit Done

2. Enter the appropriate values in the following fields:

## Using the FTF GUI

### *Stopping Monitoring*

- **Components Host Node** – Determines the node on which components are to be shut down.
  - **Number of Seconds to Wait** – Determines the number of seconds to allow the ending of the component to take place. If a component-ending message has not been received in the designated time, the end operation fails.
3. To shut down the specified components regardless of whether work is pending on them, check the **Terminate with Pending Work** check box.
  4. Select the component types you want to shut down on the specified node. To select a component type, check its check box. To deselect a component type, uncheck its check box. You can select any of the following component types:
    - Manager
    - Sender
    - Receiver
    - Logger
    - Status Daemon
  5. Click the **Submit** button to process the shut down request. Click the **Done** button to exit the window without processing the shut down request.

## Stopping Monitoring

When you stop monitoring, the details of the TDE status messages no longer display. You must stop monitoring in order to set the filter parameters.

To stop monitoring, click **Actions>Stop**.



---

# Tivoli Data Exchange Explorer

The Tivoli Data Exchange (TDE) Explorer allows you to execute TDE commands using a point-and-click interface. This chapter lists the software required to run the TDE Explorer and describes how to work with the TDE Explorer, which is provided as a snap-in to the Microsoft Management Console (MMC). It explains how to set up the TDE Explorer to manage TDE, make data-transfer requests, view the information according to your preference, and use the Scheduler for data-transfer tasks.

This chapter contains the following sections:

Section	Page
Software Requirements	260
Tivoli Data Exchange Explorer Overview	261
Using the Tivoli Data Exchange Explorer	262
Getting Started with Tivoli Data Exchange Explorer	265
Viewing Status Information Using Tivoli Data Exchange Explorer	267
Viewing the Tivoli Data Exchange Component Log Queue	270
Adding, Modifying, and Deleting Data Exit Definitions	271
Creating, Modifying, and Deleting Tivoli Data Exchange Requests	274
Making an FTFPing Request from Tivoli Data Exchange Explorer	287
Scheduler	290
Scheduling a Repeating Task	290
Scheduling a Onetime Task	299
Restarting the Scheduler	303

## Assumptions

This chapter makes the following assumptions:

- You understand how queues are used in TDE and MQSeries.
- One of the following platforms is installed on your system: Win 32. Service Pack 3 or higher needs to be installed for the NT platform.

## Software Requirements

The TDE Explorer requires certain software to be installed on your computer before it can run. The following table lists the required software and the source for obtaining it:

Conditions	Where to Get Software
If Microsoft Win 32	This is available from the Microsoft Web site: <a href="http://www.microsoft.com">http://www.microsoft.com</a>
If Microsoft Win 32 – MQSeries V 5.1 or above	This is available from the IBM MQSeries Web site: <a href="http://www.software.ibm.com/ts/mqseries/txppacs/txpm1.html">http://www.software.ibm.com/ts/mqseries/txppacs/txpm1.html</a>
Microsoft Internet Explorer 4.01 with SP 1 or later	This is available from the Microsoft Web site: <a href="http://www.microsoft.com">http://www.microsoft.com</a>
Microsoft HTML Help V1.2	This is available from the Microsoft Web site: <a href="http://www.microsoft.com">http://www.microsoft.com</a>
Microsoft Management Console V1.1	This is available from the Microsoft Web site: <a href="http://www.microsoft.com">http://www.microsoft.com</a>
TDE configuration file	Provided by your TDE administrator

The TDE Explorer is included in the TDE installation process. To start the TDE Explorer, select **Start>Programs>TDE>TDE Explorer**.

# Tivoli Data Exchange Explorer Overview

## What Is the TDE Explorer?

The TDE Explorer is provided as a snap-in to the Microsoft Management Console (MMC). MMC is a tool host that provides a common environment for snap-ins. A snap-in is a tool hosted by MMC and displayed as a console. As extensions to this tool host, snap-ins add system management functionality.

The TDE Explorer allows you to manage and administer TDE. You can use the TDE Explorer to manage an entire TDE network. Working with MQSeries, the TDE Explorer can be operated against any TDE node that MQSeries supports.

The TDE Explorer is an alternative to the standard Graphical User Interface (GUI) shipped with TDE.

The main features are:

- There is close integration with the Microsoft Win 32 environments by using the extra functionality provided by the Microsoft Management Console, Version 1.1.
- You can use the MQSeries MQI interface to connect to local or remote TDE nodes.

---

### **Note:**

If you use the TDE Explorer against an OS/390 TDE node, you will need to have the optional MQSeries Version 2.1 for OS/390 feature installed on your system.

---

The main highlights of TDE Explorer, the graphical tool for managing, controlling, and exploring TDE, are:

- A Windows Explorer-type view appears which allows you to filter, customize views of the FTF status columns, sort on the columns, perform detailed drilldown of a selected entry, and do selected printing and purging.
- You can show and hide the displayed results based on Active, Completed, or Failed status.

## **Tivoli Data Exchange Explorer**

### *Using the Tivoli Data Exchange Explorer*

- You have the ability to manage multiple TDE nodes, local or remote.
- You can issue FTFPING requests against local or remote TDE nodes.
- You can issue previously stored or dynamically created TDE requests against local or remote TDE nodes.
- You can use an entry screen to specify and store TDE exit and connector information.
- You can use an entry screen to formulate and store TDE requests.
- You can view the TDE component's runtime log queue.

## **Using the Tivoli Data Exchange Explorer**

The MMC displays a window that looks like the Windows Explorer. After adding the TDE snap-in to your MMC, you can use the console tree pane on the left to navigate and select TDE nodes. The Results pane, on the right, shows a view of the selected object within the node.

In the console tree pane, right-click a TDE object to access a menu. Here you can select your required action, such as create new, modify properties, delete, or show status details. The results of the selections are displayed in the right pane.

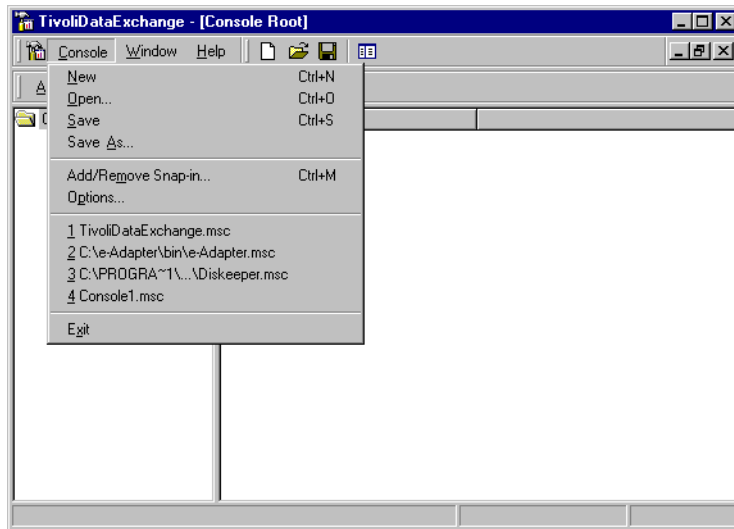
The menu differs according to the type of object and the pane. MMC functionality also allows you to use the toolbar at the top for these tasks, or the Action or View menus.

## Customizing Your Tivoli Data Exchange Explorer

You can add the TDE snap-in to your console and then save that view for use later.

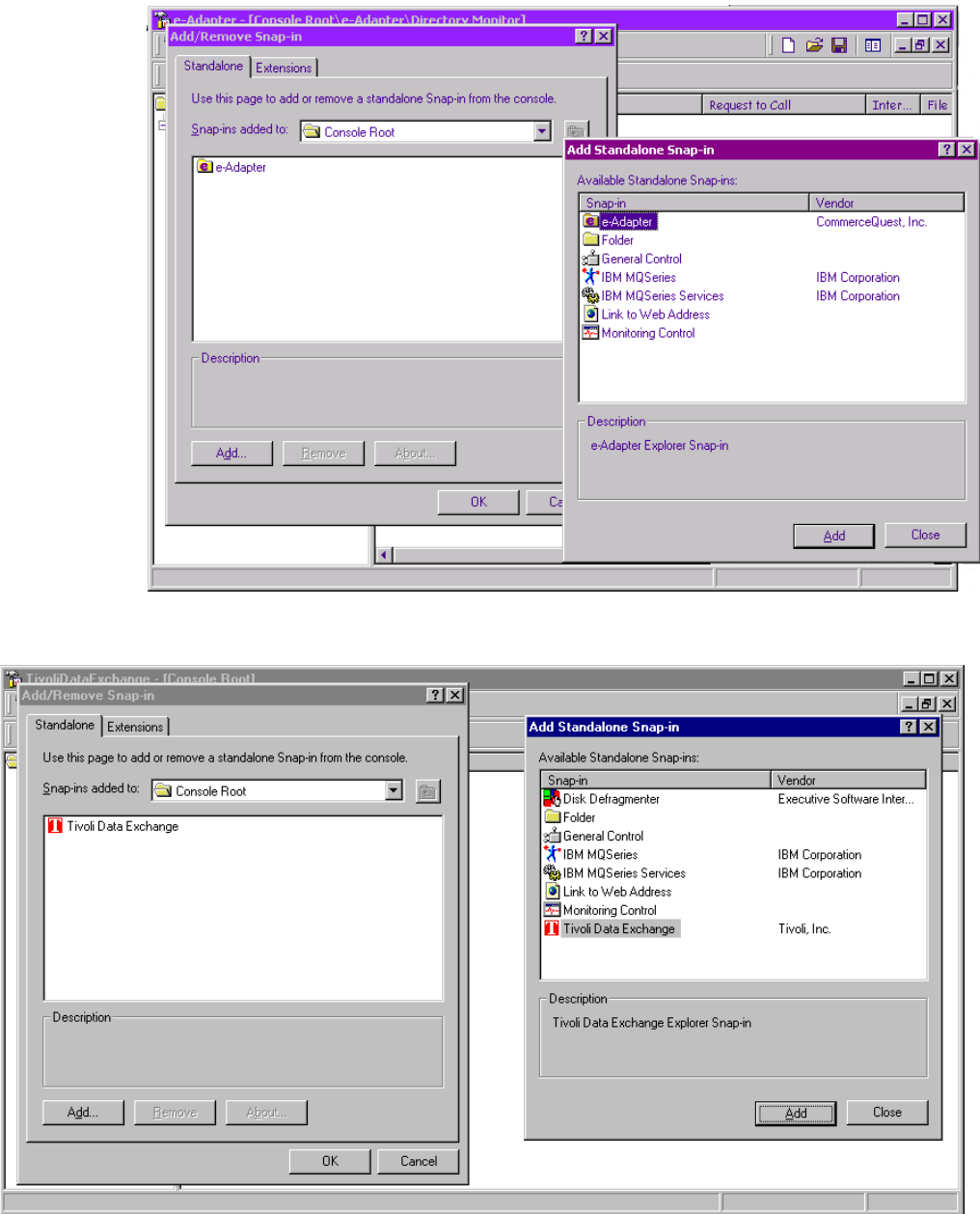
You can customize by performing the following steps:

1. On the Console menu, click **Add/Remove Snap-in**.



2. In the dialog box, click the **Add** button, select TDE, and click **Add**. Click **Add** again if you need to add another one as well. Click **OK**.

**Tivoli Data Exchange Explorer**  
*Using the Tivoli Data Exchange Explorer*



3. Set the window sizes and views to your preferences.

4. On the Console menu click **Save As** and enter a suitable filename, such as your username, and click **OK**.
5. Next time you open the TDE Explorer, on the Console menu, at the bottom, you will see your saved name with a number by it to select it and retrieve your snap-ins and view.

Some default components are installed with the TDE Explorer:

- Data Exit Definitions
- Request Macros
- TDE Nodes
- Directory Monitor
- Scheduler

Using these components is explained in the following sections after the description of how to add a node.

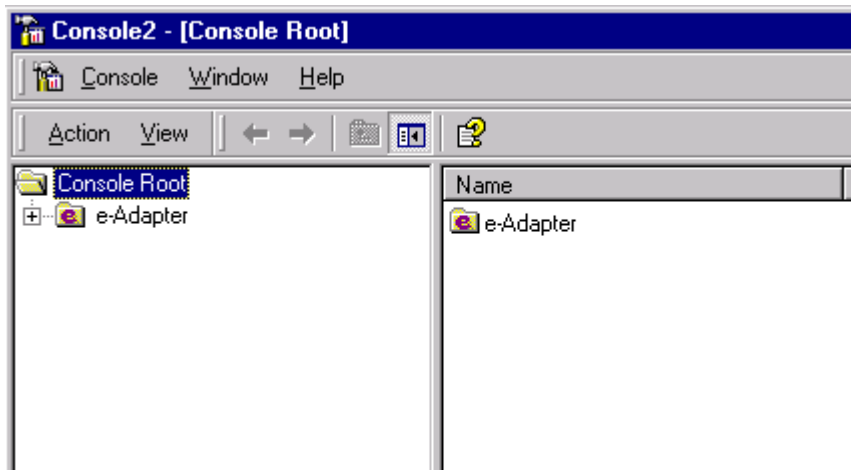
## **Getting Started with Tivoli Data Exchange Explorer**

After you have added the TDE snap-in, you can add nodes. A node represents a transfer location where you plan to send files. For example, you would create a node for each queue manager.

The starting point for TDE Explorer is the window created by adding the TDE snap-in, as described in steps 1 and 2 of “Customizing Your TDE Explorer.”

## Tivoli Data Exchange Explorer

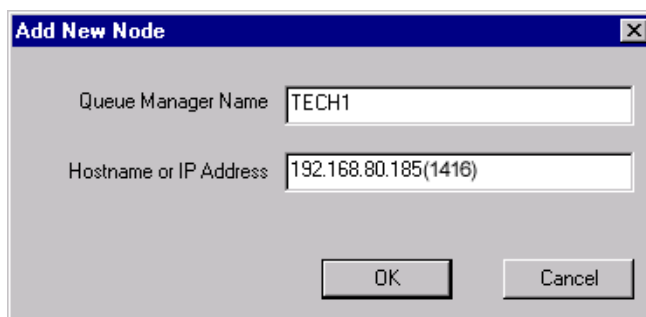
### *Getting Started with Tivoli Data Exchange Explorer*



### **Adding the First Tivoli Data Exchange Explorer Node**

Because the TDE Explorer snap-in allows you to manage multiple TDE nodes by using MQSeries, you need to provide the Queue Manager name and IP address for each node.

To specify the Queue Manager name and IP address, right-click TDE on the console tree pane of the MMC window. Choose **Add Node** from the popup menu. The Add New Node window appears:





To add a node, a TCP Listener must be active on the queue manager you are monitoring. Type the Queue Manager Name and the Hostname or IP Address in the appropriate fields. You need to ensure that the correct port number is included in the IP address. The default port number is 1414, but you can change the value as required by your environment. Enter the port number in parentheses after the IP Address if the port number is different.

---

### **Notes:**

- In order to administer the TDE node on that system, the userid from which you are using TDE must belong to MQM or the Administrators group authorized to access the Queue Manager and the TDE queues.
  - The SYSTEM.DEFAULT.SVRCONN channel must also be created to allow connection to MQSeries on remote nodes.
- 

Two options are available for each TDE Explorer node:

- Status
- Log Messages

The following sections describe how to view status and log information.

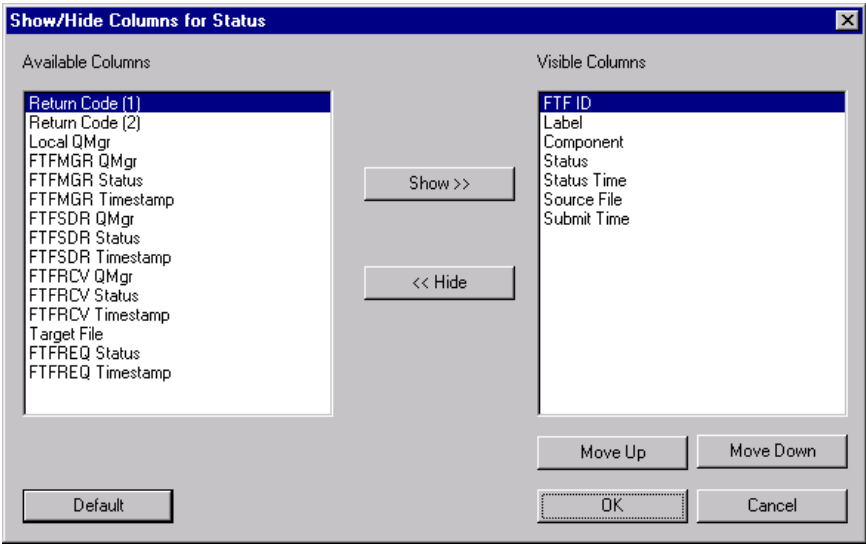
## **Viewing Status Information Using Tivoli Data Exchange Explorer**

Once you have successfully added a TDE node to the TDE Explorer, you can display TDE status information for that node. Expand the node by clicking it, and then click the **Status** node.



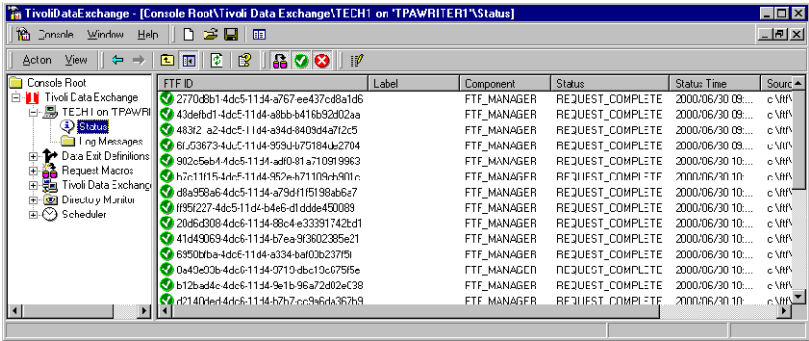
The menu bar at the top of the Console Root window contains icons that let you perform actions in one click. The **Show/Hide Status Columns** icon allows you to customize the columns that appear on the Status screen. Clicking the icon presents the following window where you can show or hide columns on the Status display.

**Tivoli Data Exchange Explorer**  
*Viewing Status Information Using Tivoli Data Exchange Explorer*



Highlight the fields you want as columns from the **Available Columns** list and click the **Show** button. To remove columns from the display, highlight columns in the **Visible Columns** list and click the **Hide** button.

When the above fields are selected to show, the following window appears.



Three different types of icons appear next to an entry in the Results pane:

*Viewing Status Information Using Tivoli Data Exchange Explorer*

- The first icon signifies that the transfer is still active. Look at the **Status** and **Component** columns to check the last known status.
- The second icon signifies that the transfer has completed successfully.
- The third icon signifies that the TDE transfer request has failed or been cancelled.

If you want complete detail about a transaction, double-click an entry in the status display. The following **Status Detail** display appears:

**2431be49-3271-11d4-8876-a20ca009a78a Properties**

Source File: TECH1 -> e:\temp\lrf.ini

Target File: TECH1 -> e:\temp\lrf.txt

Elapsed Time (DD:HH:MM:SS): 00:00:00:00

Start/Latest Time: 15:17:42 05/26/2000 - 15:17:42 05/26/2000

Current Message Number: 1

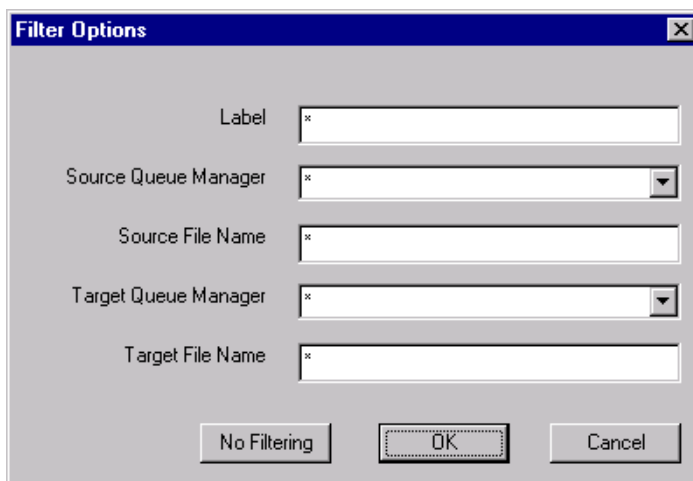
Total Message Count: 1

Group Name:

Label:

OK Cancel Apply

You can also **filter** the requests using the filtering option. Select this option by right-clicking the **Status** option for the node. Select **Set Filter Options** from the popup menu. The Filter Options window appears:

A screenshot of the 'Filter Options' dialog box. It has a title bar with 'Filter Options' and a close button. The dialog contains five input fields, each with a small 'x' icon to its left: 'Label', 'Source Queue Manager', 'Source File Name', 'Target Queue Manager', and 'Target File Name'. The 'Source Queue Manager' and 'Target Queue Manager' fields are dropdown menus. At the bottom, there are three buttons: 'No Filtering', 'OK', and 'Cancel'.

Complete the appropriate field to filter the Status view by typing information or selecting from the dropdown box, as appropriate. For example, if you want to see only the status of requests related to a particular source file, type the Source File Name in that field and click **OK**.

When filtering is active, the **Status** node has the word **(Filtering)** added to it.

## Viewing the Tivoli Data Exchange Component Log Queue

By using the TDE Explorer, you can view the TDE log file for a particular node. To view the log file:

- Double-click the node to display the options.
- Double-click the **Log Messages** option. The log queue appears in the Results pane on the right.

---

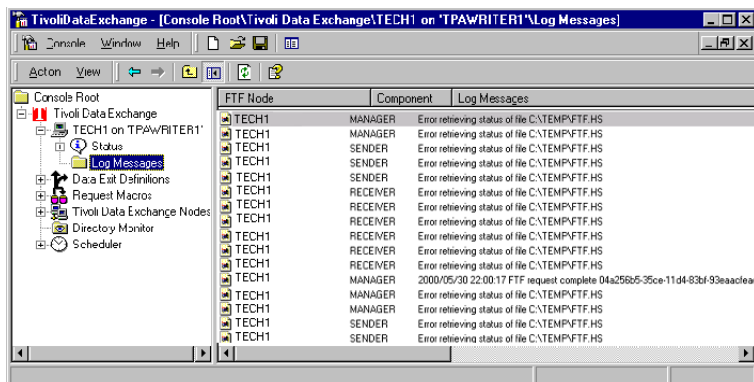
### **Note:**

Using the **Log** view on both the sending and receiving nodes can help to identify the cause of a problem.

---

## **Tivoli Data Exchange Explorer** *Adding, Modifying, and Deleting Data Exit Definitions*

The following screen shows an example of a TDE component log queue. This is only available if queue logging has been enabled via the configuration file. Sorting is enabled on both the Component and Log Messages columns.



---

### **Note:**

Some error conditions are only reported to the component log, and not to the status messages.

---

## **Adding, Modifying, and Deleting Data Exit Definitions**

Most of the power of TDE lies in the use of various preprocessing, postprocessing, and data exits. The Data Exit Definition facility within TDE Explorer facilitates the identification and specification of data exits. Using this facility, you can use the default exits provided by TDE, and you can add any exits you have written.

The TDE Explorer default exits are as follows:

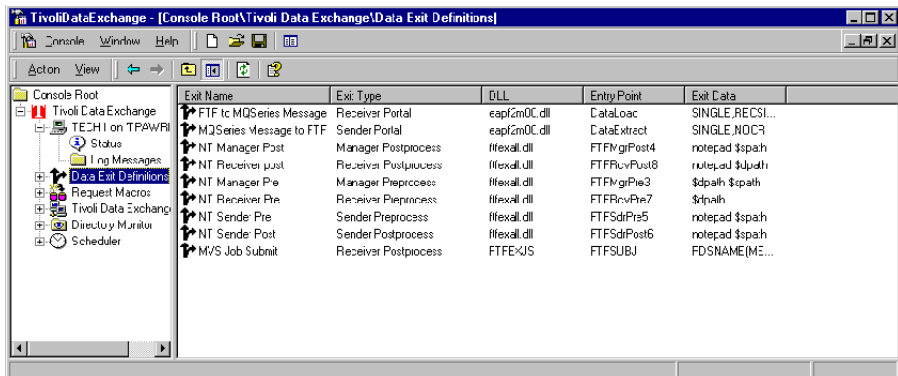
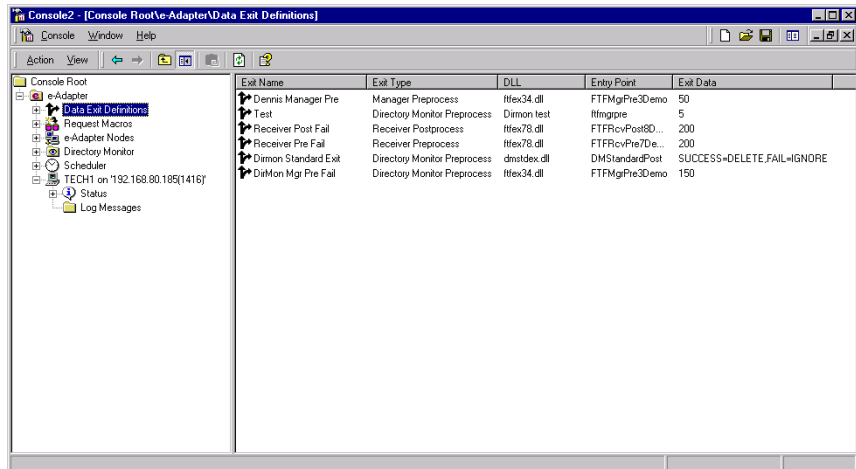
- Manager Preprocess
- Manager Postprocess
- Sender Preprocess
- Sender Postprocess
- Receiver Preprocess
- Receiver Postprocess

## Tivoli Data Exchange Explorer

### *Adding, Modifying, and Deleting Data Exit Definitions*

- Sender Connector
- Receiver Connector
- Directory Monitor Preprocess
- Directory Monitor Postprocess

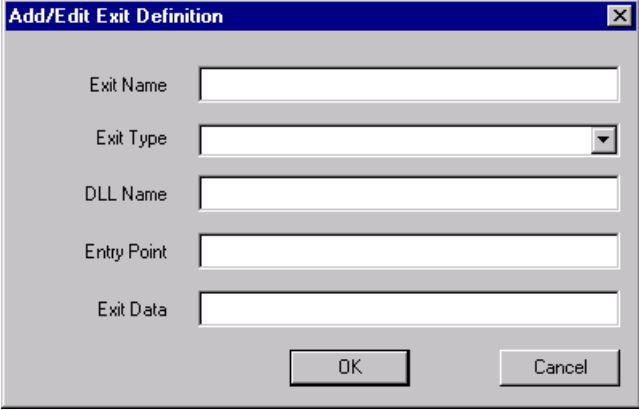
You give each exit a specific name that is assigned to a specific exit point and a specific function. This name then displays in the dropdown list of available exits when you formulate a TDE request. The available exits also show in the Results pane when you view the Data Exit Definitions option.



You add a new entry to the list of Data Exit Definitions by performing the following steps:

## **Tivoli Data Exchange Explorer** *Adding, Modifying, and Deleting Data Exit Definitions*

1. Right-click the **Data Exit Definition** component for the node and select **New>Exit Definition**. The Add/Edit Exit Definition window appears:

A screenshot of the 'Add/Edit Exit Definition' dialog box. The dialog has a title bar with the text 'Add/Edit Exit Definition' and a close button. Inside, there are five labeled text input fields: 'Exit Name', 'Exit Type' (which is a dropdown menu), 'DLL Name', 'Entry Point', and 'Exit Data'. At the bottom right, there are two buttons: 'OK' and 'Cancel'.

2. Enter the new Exit Definition by completing the fields in the window and click **OK**. Enter the following information in the fields:
  - **Exit Name** – Contains the name used to refer to this exit definition.
  - **Exit Type** – Contains the exit at which this definition is executed. Existing exit types can be selected from the pulldown list.
  - **DLL Name** – Contains the name of the module to be executed when this exit definition is invoked.
  - **Entry Point** – Contains the name where execution is to begin in the DLL.
  - **Exit Data** – Contains the parameters that are to be passed to the DLL.

To modify an existing Data Exit Definition, access it as described above, then change the properties as appropriate. Click **OK** to save the new definition. You can change the properties of a definition and keep the same name, or you can change the properties and give it a new name.

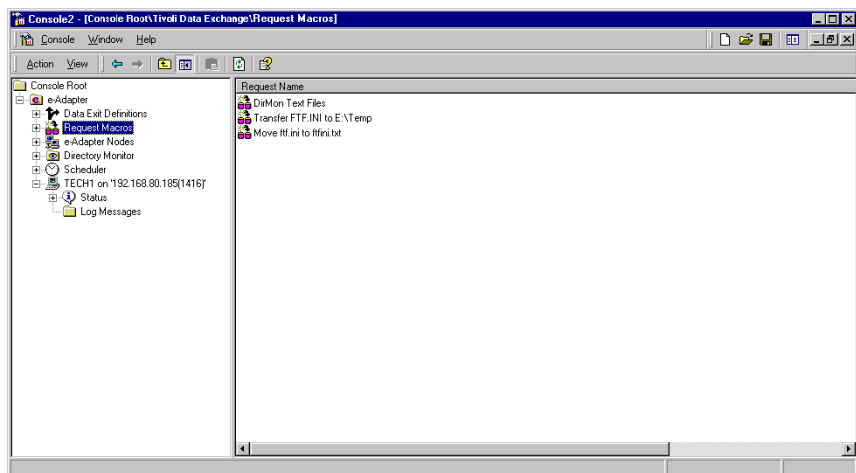
To delete a Data Exit Definition, view existing definitions in TDE Explorer by selecting **Data Exit Definitions** in the Console Tree pane. The existing definitions appear in the Results pane on the right. Right-click the name of the definition you want to remove and click **Delete**.

## Creating, Modifying, and Deleting Tivoli Data Exchange Requests

TDE requests can be confusing and complicated, particularly when they have many arguments or when you need to define the user exits, the OS/390 options, or the AS/400 options. With TDE Explorer, you have a facility to assist you in creating requests. You can create a request in one of two ways: via the Request Macros option or via a node's Task option.

### Using the Request Macros Node to Create a Request

Using the Request Macros node, right-click **Request Macros** and select **New - Request Definition**.



The New Request Properties window appears.

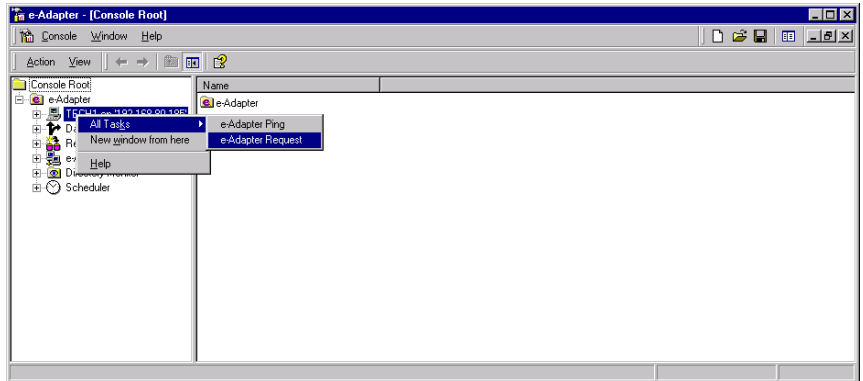
### Using a Node's Task Option to Create a Request

Using a node's Task option, right-click the node name, then select **All Tasks** and click **TDE Request**. The Select Request window appears. Click the **New Request** button.

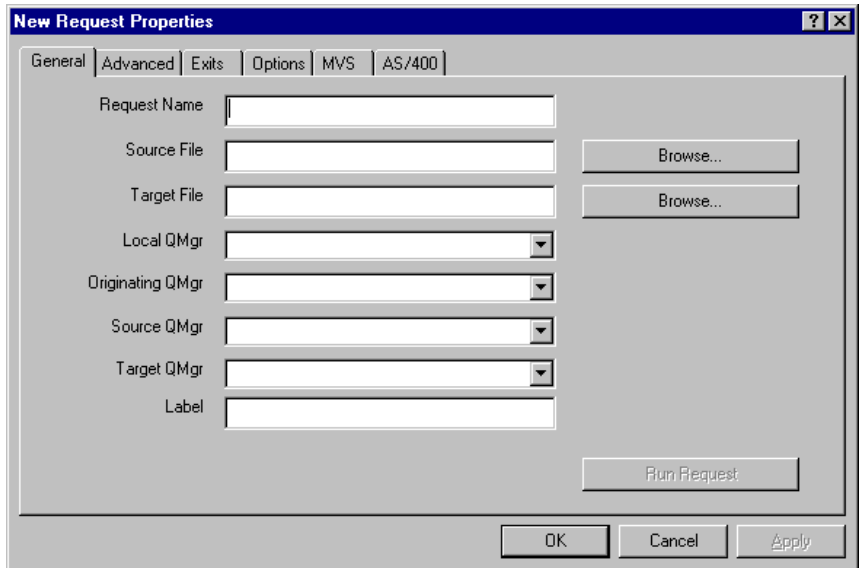


## Tivoli Data Exchange Explorer

### *Creating, Modifying, and Deleting Tivoli Data Exchange Requests*



The New Request Properties window appears.



Enter the following information to create a TDE file transfer request. After you have entered the information, click the **Run Request** button to activate the request.

#### The General Tab

- **Request Name** – Designates the name given to the request. This name is reported in the logs about the transfer.
- **Source File** – Designates the fully qualified source path and filename used in the file transfer. Use the **Browse** button to search for the file and select the appropriate file from the dialog boxes.
- **Target File** – Designates the fully qualified path and filename where the transferred information is to reside. Use the **Browse** button to search for the file and select the appropriate file from the dialog boxes.
- **Local QMgr** – Contains the name of the queue manager from which the FTF Request command is issued. This value is required on OS/390 systems. On all other systems, if it is not specified, the command connects to the default queue manager that is set in the MQSeries configuration.
- **Originating QMgr** – Contains the name of the queue manager from which the file transfer request originates. Click the down arrow to get a list of the current queue managers and select one.
- **Source QMgr** – Contains the name of the queue manager from which the source data is transferred. Click the down arrow to get a list of the current queue managers and select one.
- **Target QMgr** – Contains the name of the queue manager to which the data associated with the file transfer request is written. Click the down arrow to get a list of the current queue managers and select one.
- **Label** – User-defined information to describe the file-transfer request. The status reporting system displays the contents of this field.

## The Advanced Tab

The screenshot shows the 'New Request Properties' dialog box with the 'Advanced' tab selected. The dialog has a title bar with a question mark and close button. Below the title bar are tabs: 'General', 'Advanced', 'Exits', 'Options', 'MVS', and 'AS/400'. The 'Advanced' tab contains several input fields and two grouped sections. The fields are: 'Source Type', 'Source Data', 'Destination Type', 'Destination Data', 'Pool', 'ID 1', 'ID 2', and 'ID 3'. The 'Reply Options' group contains 'Reply Queue' and 'Reply Queue Manager'. The 'Notify Options' group contains 'Notify Status' (a dropdown menu currently showing 'NONE'), 'Notify Type', and 'Notify Data'. At the bottom right are 'OK', 'Cancel', and 'Apply' buttons.

Click the **Advanced** tab and enter values in the following fields as necessary.

- **Source Type** – Determines the data type for the source data. You should only use this argument to handle source data that is not stored in a file. This value must match a data type specified in the TDE configuration file, and it is case sensitive.
- **Source Data** – Determines the data input for a transfer that sends data that is not stored in a file.
- **Destination Type** – Determines the data type for the destination data. You should only use this argument to handle destination data that is not stored in a file. This value must match a data type specified in the TDE configuration file, and it is case sensitive.
- **Destination Data** – Determines the data output for a transfer that receives data that is not stored in a file
- **Reply Queue** – Names the queue to which reply messages are to be routed.

## Tivoli Data Exchange Explorer

### *Creating, Modifying, and Deleting Tivoli Data Exchange Requests*

- **Reply Queue Manager** – Names the queue manager to which reply messages are to be routed.
- **Pool** – Contains the name of the data pool used for transferring data from the TDE Sender to the TDE Receiver. If this value is not specified, the default pool specified in the TDE configuration file is used. For this option to function, the specified pool must be defined in the TDE configuration file.
- **ID 1** – Designates the first user-defined field that is associated with a data-transfer. The identifierText1 field can contain any value that you desire. If the identifier text value contains spaces, the value needs to be in quotes. The value placed in the field is carried with the data-transfer and the values are available to you in exit routines. Additionally, the FTFSTAT command displays the values of the identifier fields when you request a status display.
- **ID 2** – Designates the second user-defined field that is associated with a data-transfer. The identifierText2 field can contain any value that you desire. If the identifier text value contains spaces, the value needs to be in quotes. The value placed in the field is carried with the data-transfer and the values are available to you in exit routines. Additionally, the FTFSTAT command displays the values of the identifier fields when you request a status display.
- **ID 3** – Designates the third user-defined field that is associated with a data-transfer. The identifierText3 field can contain any value that you desire. If the identifier text value contains spaces, the value needs to be in quotes. The value placed in the field is carried with the data-transfer and the values are available to you in exit routines. Additionally, the FTFSTAT command displays the values of the identifier fields when you request a status display.
- **Notify Status** – Defines when a notification message will be sent to the NotifyQueue (see *TDE Installation Guide*, “TDE Configuration,” for more information). The notification is sent if the transaction’s status matches the status specified in this argument. TDE delivers notification messages, but does not perform any further processing. If you specify this argument, you must also specify *Notify Data* and *Notify Type* arguments. **Valid values:** Success, Failure, Nonsuccess (includes failed, cancelled, expired)

## Tivoli Data Exchange Explorer

### *Creating, Modifying, and Deleting Tivoli Data Exchange Requests*

- **Notify Type** – Specifies the user-defined method, such as e-mail, pager, or fax, that will be used to deliver a notification message based on a transaction's status. If you specify this argument, you must also specify *Notify Data* and *Notify Status* arguments.
- **Notify Data** – Specifies user-defined data to aid in notification, such as e-mail, pager, or fax information, that will be used to deliver a notification message based on a transaction's status. If you specify this argument, you must also specify *Notify Status* and *Notify Type* arguments.

### The Exits Tab

Exit Name	Value	Exit Data
Manager Pre	None	
Manager Post	None	
Sender Pre	None	
Sender Post	None	
Receiver Pre	None	
Receiver Post	None	
Sender Portal	None	
Receiver Portal	None	

Click the **Exits** tab and enter values in the following fields as necessary.

- **Manager Pre** – If specified, this is the first exit executed in the data-transfer request.
- **Manager Post** – If specified, this is the last exit executed in the data-transfer request.
- **Sender Pre** – This exit is invoked before the TDE Sender reads the source file's contents.

## Tivoli Data Exchange Explorer

### *Creating, Modifying, and Deleting Tivoli Data Exchange Requests*

- **Sender Post** – The TDE Sender invokes this exit after the source file's contents have been read and processed for transfer to the target node.
- **Receiver Pre** – The TDE Receiver invokes this exit before it begins to write the target file.
- **Receiver Post** – The TDE Receiver invokes this exit after the target file has been processed and all the source data has been written to it.
- **Sender Connector** – Invokes the Sender connector exit.
- **Receiver Connector** – Invokes the Receiver connector exit.
- **Exit Data** – Contains the command-line argument to execute when you invoke a user exit that requires input parameters.

### The Options Tab

The screenshot shows the 'New Request Properties' dialog box with the 'Options' tab selected. The dialog has a title bar with a question mark and close button. Below the title bar are tabs: 'General', 'Advanced', 'Exits', 'Options' (selected), 'MVS', and 'AS/400'. The 'Options' tab contains three main sections: 'Performance/Recoverability', 'Staging options', and 'Transfer'. In 'Performance/Recoverability', 'Message Size (k)' is 512, 'Expiry (minutes)' is 0, 'Priority' is 3, and checkboxes for 'Immediate', 'Trusted', 'Compress', and 'Persist Data' are present. In 'Staging options', checkboxes for 'Stage and submit', 'From stage', 'Stage only', and 'Persist Stage' are present, along with an 'FTF ID' field. In 'Transfer', 'Write Mode' is CREATE, 'Wrap Mode' is None, 'Cancel Mode' is Default, and checkboxes for 'Binary', 'Delete Source', 'Pad Records', 'Create Directory', 'DBCS', and 'Pad Character' are present. At the bottom are 'OK', 'Cancel', and 'Apply' buttons.

Click the **Options** tab and enter values in the following fields as necessary.

- **Message Size** – Allows you to set a message size value to override the MQSeries message size value. **Valid values:** 1-3906 (3.9 MB) **Default value:** 512

- **Expiry** – Determines the time period, measured in minutes, after which the data-transfer request expires. If the expiration duration is exceeded, the request is terminated and the FTFRCL\_REQUEST\_EXPIRED message is returned. If the expiration occurs partway through a request, the request is marked as expired.
- **Priority** – Determines the priority applied to the data-transfer request.  
**Valid values:** 1 (highest) – 5 (lowest) **Default value:** 5
- **Immediate** – Gives you the ability to issue a TDE data-transfer request that is processed synchronously between the Sender and Receiver rather than the normal asynchronous mode. Using this argument differentiates an immediate data-transfer from a regular TDE transfer.
- **Trusted** – Sacrifices the ability for recovery in order to provide greater performance. In a trusted transaction, no file recovery is possible. Specifying this argument invokes the TDE trusted option, not the MQSeries trusted option.
- **Compress** – Specifies that the data being sent is compressed using the TDE default compression algorithm.
- **Persist Data** – Specifies that the data being transferred is persistent. If you specify this argument, the data still exists after a system or TDE reboot or shutdown. Selecting this argument increases the recovery ability of TDE but reduces performance.
- **Stage and submit** – Enables the data messages that make up the data being transferred to be stored in a staging area and remain there after the data-transfer transaction has ended.
- **Stage only** – Places the data messages on the staging queue but does not send the messages. If you specify this value, you cannot specify a destination queue manager or destination file value.
- **Persist Stage** – Specifies that the messages in the staging area are persistent. If you specify this argument, the messages still exist after a system or reboot or shutdown. Selecting this argument increases the recovery ability of TDE but reduces performance.
- **From Stage** – Indicates that the source file specified in the *-spath* argument should be retrieved from the staging queue rather than from the actual source file.

## Tivoli Data Exchange Explorer

### *Creating, Modifying, and Deleting Tivoli Data Exchange Requests*

- **FTF ID** – Indicates the FTFID of the transaction that should be sent from the staging queue. You should not use this option unless you are sending a file from the staging queue using the *-fromstage* option, and you have not specified a source file name.
- **Write Mode** – Determines what occurs when the data is written to the target. Although the default value for this argument is *create*, the only values you can specify from the command line are *append* and *noreplace*. Do not use this argument unless you want to override the default *create* setting. **Valid values:** APPEND, CREATE, NOREPLACE **Default value:** CREATE
- **Wrap Mode** – Indicates how records will be processed when they reach the target file and the records are longer than the target record length. **Valid values:** None, WRAP, TRUNCATE, FAIL
- **Cancel Mode** – Provides a command-line override to the preemptive cancel flag in the TDE configuration file.
- **Binary** – Designates the file type to be binary.
- **Create Directory** – Creates the directories required to support the **destPath** value. If this argument is not specified and the specified directory does not exist, the data-transfer request fails with a FILE OPEN ERROR.
- **Delete Source** – Indicates that the source data is to be deleted once the data-transfer request is completed.
- **DBCS** – Indicates the transferred data is stored using the double-byte character set.
- **Pad Records** – Enables or disables the padding facility.
- **Pad Character** – If padding of target file records is elected by entering the argument of `recpad=pad`, the `padCharacter` indicates what hexadecimal character is to be used for padding. If left out of the command and padding is elected, blank is assumed to be the padding character.



## The MVS Tab

The screenshot shows the 'New Request Properties' dialog box with the 'MVS' tab selected. The dialog has a title bar with a question mark and close button. Below the title bar are tabs: General, Advanced, Exits, Options, MVS, and AS/400. The MVS tab is active, showing several input fields arranged in two columns. The left column contains: Organization (dropdown menu with 'None' selected), Record Format (dropdown menu with 'None' selected), Record Length (text box with '0'), Block Size (text box with '0'), Allocation Unit (dropdown menu with 'None' selected), Number of I/O Buffers (text box with '0'), and PDS Directory Blocks (text box with '0'). The right column contains: Esoteric Unit Name (text box), Primary (text box with '0'), Secondary (text box with '0'), Volume Serial # (text box), and Model Dataset (text box). At the bottom right are three buttons: OK, Cancel, and Apply.

Click the **MVS** tab and enter values in the following fields as necessary.

- **Organization** – Determines the file organization of the target file on OS/390. This argument is not required for a preallocated data set. **Valid values:** Physical Sequential (PS), Partitioned Data Set (PDS)
- **Record Format** – Determines the record format for the target file on OS/390. **Valid values:** F (fixed), V (variable), FB (fixed block), and VB (variable block)
- **Record Length** – Determines the logical record length, defined in bytes, for the target file on OS/390. If the value for recfmt is V or VB, then the value for lrecl should be 4 bytes greater than the longest data record. **Valid values:** 1-32760

## Tivoli Data Exchange Explorer

### *Creating, Modifying, and Deleting Tivoli Data Exchange Requests*

- **Block Size** – Determines the block size, defined in bytes, for the target file on OS/390. Specifying a block size of 0 enables the system to choose the optimum block size for the data set during allocation. If the record format is Fixed Block (FB), the block size in the blksize argument must be a multiple of the logical record length, the lrecl parameter. When the record format is Variable Block (VB), the blksize value must be at least four bytes greater than the lrecl value. **Valid values:** 0 - 32760
- **Allocation Unit** – Determines the allocation unit used for the target on OS/390. **Valid values:** CYL (cylinder), BLK (block), and TRK (track)
- **Number of I/O Buffers** – Allows you to specify the number of internal buffers to be used when processing data-transfers. The throughput of a TDE data-transfer is governed by a combination of the block size of the data being transferred and the number of buffers allocated for transfer in the bufno argument. **Valid Values:** 1 - 255
- **PDS Directory Blocks** – Sets up the number of directory blocks used to allocate the target PDS if it does not exist. If this argument is not specified, the value in the configuration file is used. If neither is specified, the PDS allocation will fail. **Valid values:** 1-32760
- **Esoteric Unit Name** – A symbolic name used to reference a group or class of hardware devices in an OS/390 environment.
- **Primary** – Determines the number of primary allocation units required on OS/390.
- **Secondary** – Determines the number of secondary allocation units required on OS/390.
- **Volume Serial #** – Determines the volume serial number for the target on OS/390. This argument's value is installation-dependent. Obtain it from your OS/390 System Administrator.
- **Model Dataset** – Indicates a model data sets for Generation Data Group (GDG) allocation. Consult your OS/390 System Administrator for the available model data sets.

## The AS/400 Tab

The screenshot shows the 'New Request Properties' dialog box with the 'AS/400' tab selected. The dialog has a title bar with a question mark and a close button. Below the title bar are tabs for 'General', 'Advanced', 'Exits', 'Options', 'MVS', and 'AS/400'. The 'AS/400' tab contains the following fields:

- File Type**: A dropdown menu currently set to 'None'.
- Library Auxiliary Storage Pool**: A text input field containing '0'.
- File Auxiliary Storage Pool**: A text input field containing '0'.
- Library Text**: A text input field.
- File Text**: A text input field.
- Coded Character Set**: A text input field.
- Record Length**: A text input field containing '0'.
- Create Library**: An unchecked checkbox.

At the bottom right of the dialog are three buttons: 'OK', 'Cancel', and 'Apply'.

Click the **AS/400** tab and enter values in the following fields as necessary.

- **File Type** – The value entered specifies the type of file. **Valid values:**
  - \*DFLT (defaults to the value entered in the configuration table)
  - \*SAVE (specifies an AS/400 Save file)
  - \*SRCPF (specifies an AS/400 source physical file)
- **Library Auxiliary Storage Pool** – Specifies the Library Auxiliary Pool for a library that TDE creates for a data-transfer request. **Valid values:** 1-16
- **File Auxiliary Storage Pool** – Specifies the File Auxiliary Pool for a file that TDE creates for a data-transfer request. **Valid values:** 1-16
- **Library Text** – Specifies the library description for a library that TDE creates for a data-transfer request.
- **File Text** – Specifies the file description for a library that TDE creates for a data-transfer request.
- **Coded Character Set** – The CCSID is used as the identifier for the data-transfer request. If CCSID is not specified, TDE uses the CCSID of the job. **Valid values:** 1-65535

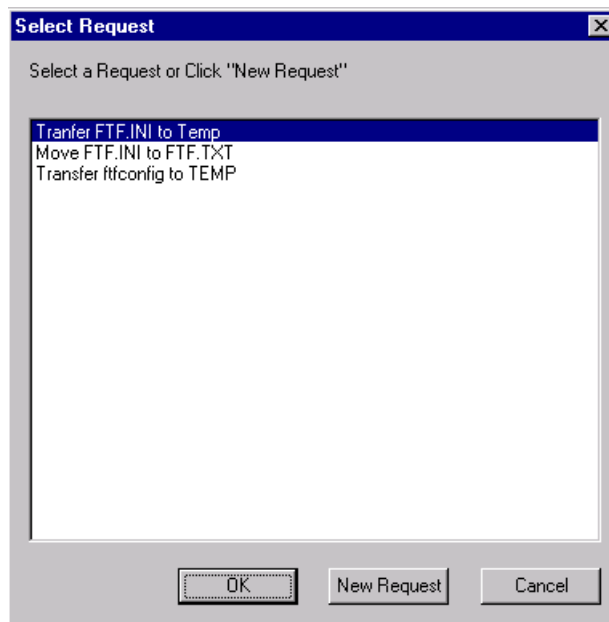
## Tivoli Data Exchange Explorer

### *Creating, Modifying, and Deleting Tivoli Data Exchange Requests*

- **Record Length** – Determines the record length for the target file on OS/390. **Valid values:** 13-32766
- **Create Library** – Specifies that TDE is to create the specified library if it does not exist.

### **Modifying, Deleting, and Running Existing Requests**

Once requests are saved, they are stored into the registry. The requests are made available to you when you select a TDE node and right-click the node name, then select **All Tasks** and click **TDE Request**. Highlight the request you want to access and click **OK**.

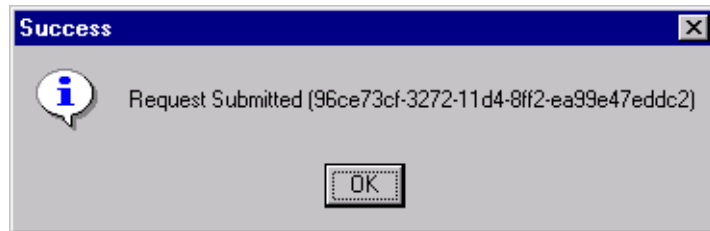


The New Request Properties window appears with all the information entered for the request you selected. To modify the request, change the properties as appropriate. Click **OK** to save the new request. You can change the properties of a request and keep the same name, or you can change the properties and give it a new name.

*Making an FTFPing Request from Tivoli Data Exchange Explorer*

To delete a request, view existing requests in TDE Explorer by selecting **Request Macros** in the Console Tree pane. The existing requests appear in the Results pane on the right. Right-click the name of the request you want to remove and click **Delete**.

Remember that once you select a request, you can still change any of the options. When ready, click the **Run Request** button to execute the request. If the request is accepted, the Success window appears.



Click **OK** and you return to the **TDE Status** node. Click the **Refresh** icon on the menu bar to get an updated status screen.

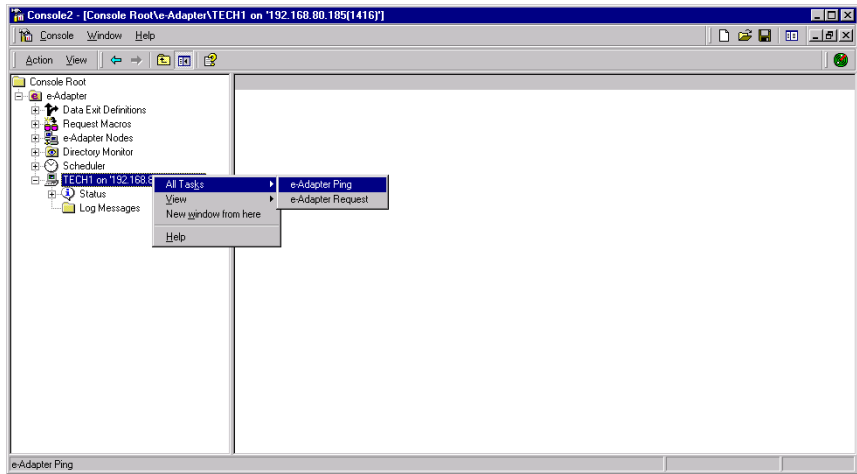
## **Making an FTFPing Request from Tivoli Data Exchange Explorer**

FTFPing is an excellent facility to verify the availability of the various components of TDE. It also helps check network performance. To access the FTFPing window, start at the node on the Console Root.

- Right-click the node name to get the small pop-up window.
- Click the **All Tasks** option.
- Click **TDE Ping**.

## Tivoli Data Exchange Explorer

### *Making an FTFPing Request from Tivoli Data Exchange Explorer*



The Ping window appears. Enter the values in the fields and click the **Ping** button.

## Tivoli Data Exchange Explorer

### *Making an FTFPing Request from Tivoli Data Exchange Explorer*

The **e-Adapter Ping** dialog box contains a "Ping Settings" section with the following fields:

- Local Queue Manager:
- Originating Queue Manager:
- Source Queue Manager:
- Destination Queue Manager:
- Number of Seconds to Wait:
- Ping Priority:
- Message Size (K):

Below the settings is a text area displaying the results of the ping operation:

```
TECH1->TECH1->TECH1->TECH1 bytes=310 time<=0 secs  
TECH1->TECH1->TECH1->TECH1 bytes=310 time<=0 secs  
TECH1->TECH1->TECH1->TECH1 bytes=310 time<=1 secs  
TECH1->TECH1->TECH1->TECH1 bytes=310 time<=0 secs  
TECH1->TECH1->TECH1->TECH1 bytes=310 time<=0 secs
```

On the right side of the dialog are three buttons: **Ping**, **Clear**, and **Cancel**.

The **Ping** dialog box contains a "Ping Settings" section with the following fields:

- Local Queue Manager:
- Originating Queue Manager:
- Source Queue Manager:
- Destination Queue Manager:
- Number of Seconds to Wait:
- Ping Priority:
- Message Size (K):

Below the settings is a large empty text area for displaying results. On the right side of the dialog are three buttons: **Ping**, **Clear**, and **Cancel**.

The results of the FTFPing are displayed in the window pane at the bottom. As you can see in the example above, you can also adjust the size of the ping message.

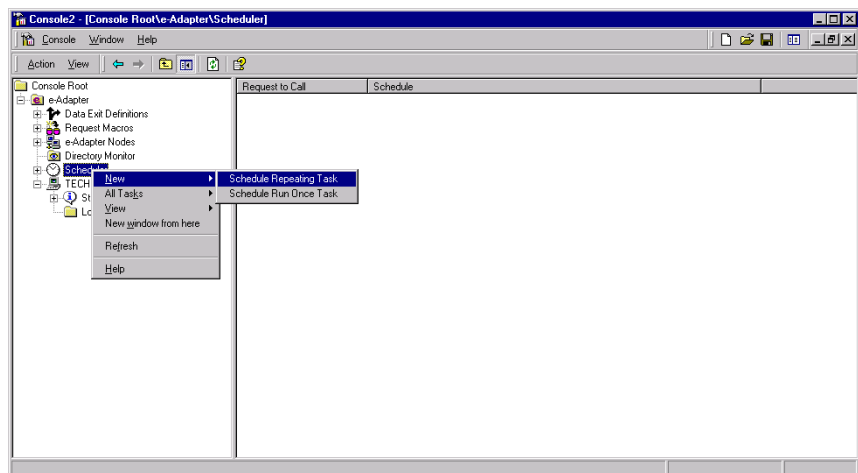
## Scheduler

The Scheduler allows you to set data transfers as a onetime task or have a task repeated. Depending on your requirements, you may have data transfers that need to occur at a certain time of day, week, or month. You can set up a Scheduler task to perform a data transfer as an output becomes available from your system. The task could be a onetime only or it might be that you want the data transfer to occur each time the data becomes available.

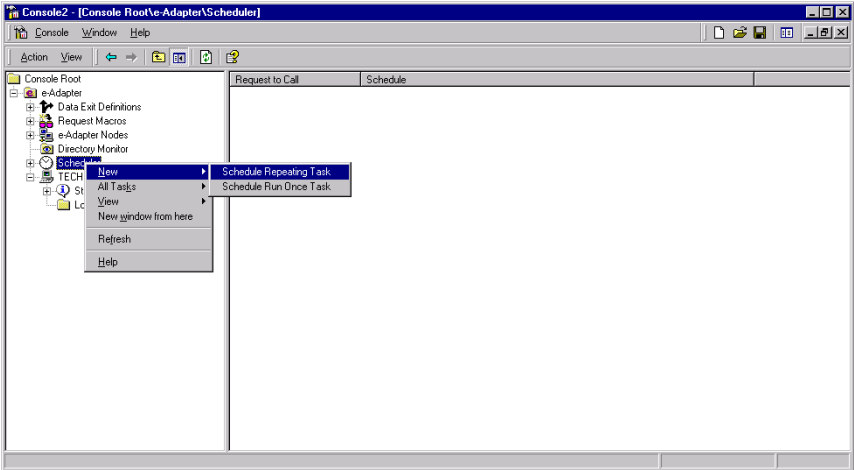
Scheduled tasks that you have set up can easily be cancelled or changed. The interface screens allow you to edit a task by double-clicking the task in the Scheduler view. Scheduler presents the base screen describing the task. You can then click the customized portions of your task and make the necessary changes. Scheduled tasks can be deleted at any time.

## Scheduling a Repeating Task

The Scheduler is an installed part of the TDE Explorer snap-in and appears as part of the list of components when you click TDE. Highlight the Scheduler by clicking the entry, then right-click the highlighted entry to access the pull-down menu. Select **New>Schedule a Repeating Task** to begin the process.



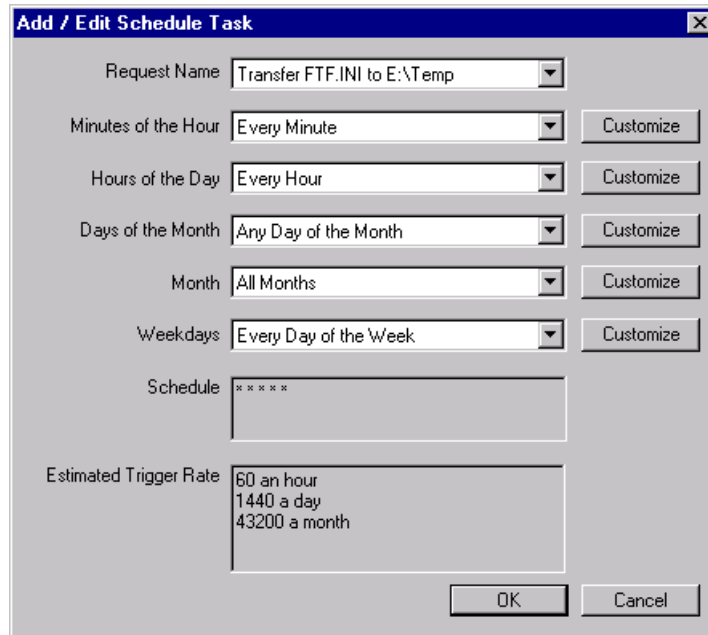




## Tivoli Data Exchange Explorer

### *Scheduling a Repeating Task*

The **Add/Edit Schedule Task** screen appears. If you are adding a scheduled task, you enter the characteristics that you want. If you are editing a task, you can change any of the characteristics.



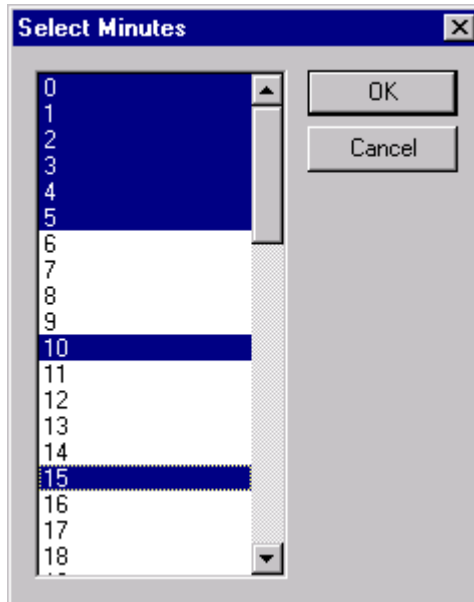
The dialog box titled "Add / Edit Schedule Task" contains the following fields and controls:

- Request Name:** A dropdown menu with the text "Transfer FTF.INI to E:\Temp".
- Minutes of the Hour:** A dropdown menu with "Every Minute" and a "Customize" button.
- Hours of the Day:** A dropdown menu with "Every Hour" and a "Customize" button.
- Days of the Month:** A dropdown menu with "Any Day of the Month" and a "Customize" button.
- Month:** A dropdown menu with "All Months" and a "Customize" button.
- Weekdays:** A dropdown menu with "Every Day of the Week" and a "Customize" button.
- Schedule:** A text box containing "x x x x x".
- Estimated Trigger Rate:** A text box containing "60 an hour", "1440 a day", and "43200 a month".
- Buttons:** "OK" and "Cancel" buttons at the bottom right.

Enter the following values in the fields as you need to schedule a task.

- **Request Name** – You may either click the arrow to get a dropdown list of names from which you can select or enter a new name. These are valid Request Macro names that have been previously entered. See “Creating, Modifying, and Deleting Tivoli Data Exchange Requests” on page 274.

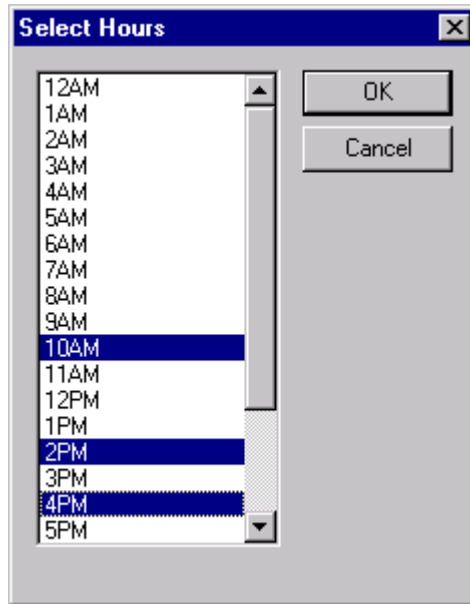
- **Minutes of the Hour** – You can click the arrow and select from the list or you can click the **Customize** button and the following screen displays. Scroll through the list and highlight the minutes on which you want the task to occur. When you have selected all desired minutes, click the **OK** button.



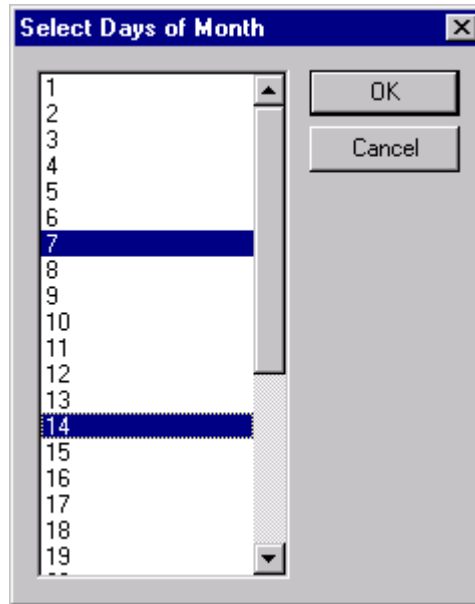
## Tivoli Data Exchange Explorer

### *Scheduling a Repeating Task*

- **Hours of the Day** – Click the arrow and select from the list or you can click the **Customize** button and the following screen appears. Scroll through the list and highlight the hours on which you want the task to occur. When you have selected all desired hours, click **OK**.



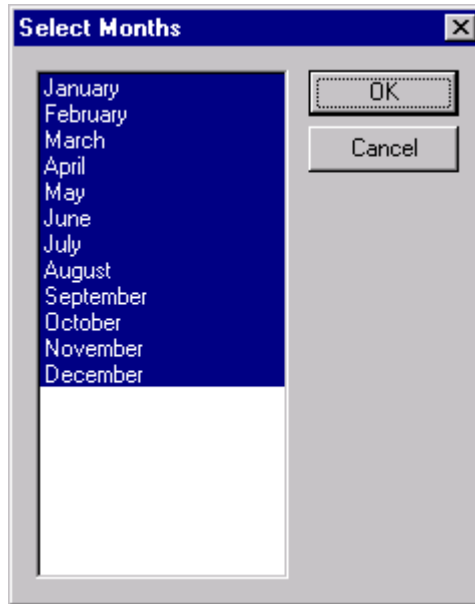
- **Days of the Month** – Click the arrow and select from the list or click the **Customize** button and the following screen appears. Scroll through the list and highlight the days of the month on which you want the task to occur. When you have selected all desired days, click **OK**.



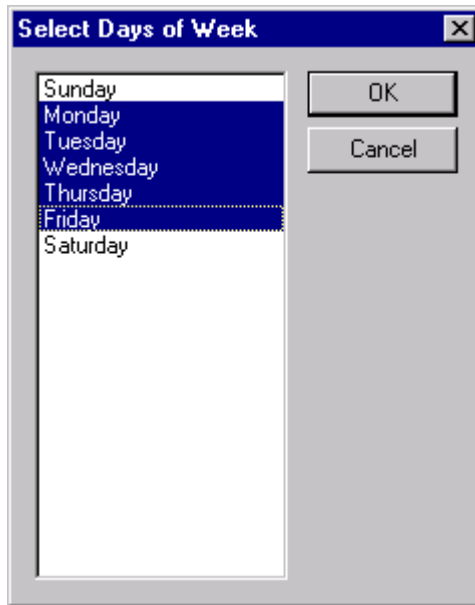
## Tivoli Data Exchange Explorer

### *Scheduling a Repeating Task*

- **Month** – Click the arrow and select from the list or click the **Customize** button and the following screen appears. Scroll through the list and highlight the months on which you want the task to occur. When you have selected all desired months, click **OK**.



- **Weekdays** – Click the arrow and select from the list or click the **Customize** button and the following screen appears. Scroll through the list and highlight the days of the week on which you want the task to occur. When you have selected all desired days of the week, click **OK**.



## Tivoli Data Exchange Explorer

### *Scheduling a Repeating Task*

After you click **OK** on each of the custom screens, you are returned to the screen below. The characteristics of the task that you selected are recorded. If you click the **Customize** buttons, your selections would be retrieved.

The Scheduler computes some information for you and displays the results in the two areas at the bottom of the window.

- **Schedule** – Contains all of the entries that you have recorded with each of the **Customize** button entries.
- **Estimated Trigger Rate** – Displays the computed estimated times that a task would operate per hour, day, and month.

If all appears to be as you desire it, click the **OK** button. If you do not want this task recorded, click the **Cancel** button.

**Add / Edit Schedule Task**

Request Name: Transfer FTF.INI to E:\Temp

Minutes of the Hour: (Custom) **Customize**

Hours of the Day: (Custom) **Customize**

Days of the Month: (Custom) **Customize**

Month: All Months **Customize**

Weekdays: Monday through Friday **Customize**

Schedule: 0,1,2,3,4,5,10,15 10,14,16 7,14 \* 1,2,3,4,5

Estimated Trigger Rate: 8 an hour (but only 3 hours a day)  
24 a day (but only 2 days a month)  
480 a month

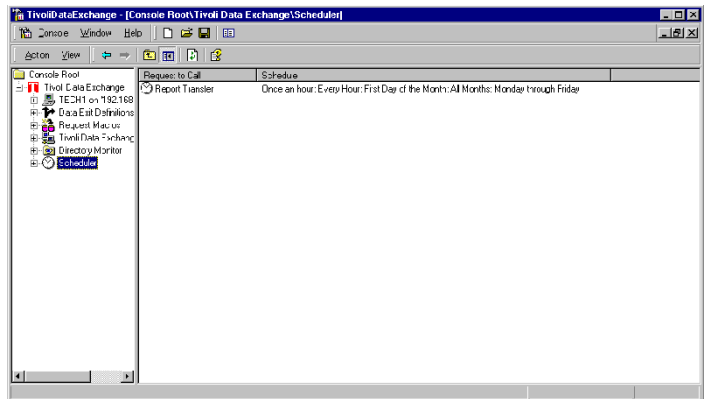
**OK** **Cancel**



When you click **OK** on the **Add/Edit Schedule Task** window, the following window appears. If this is the only task that you are going to schedule and you want it to take effect, click the **Yes** button. If you want to schedule other tasks, click the **No** button. When you have entered all of the scheduled tasks, you need to restart the Directory Monitor Service for them to take effect.



When the Directory Monitor Service is restarted, the scheduled task is displayed in the Scheduler window.

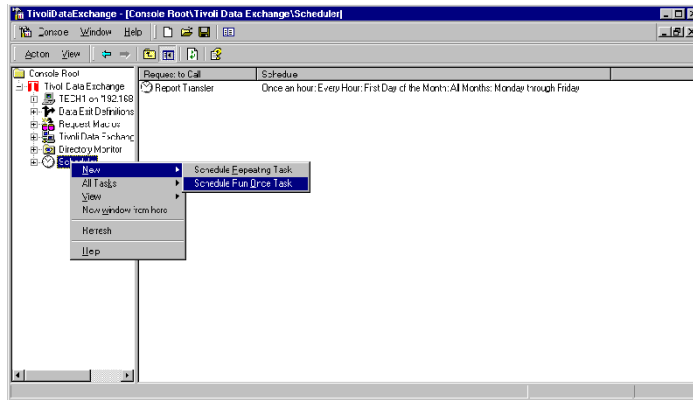


## Scheduling a Onetime Task

In this case, you want to schedule a task that will occur once. Start with the Scheduler view of TDE Explorer. Right-click **Schedule** to get the pull-down menu. Highlight **New>Schedule Run Once Task** and click the entry.

## Tivoli Data Exchange Explorer

### *Scheduling a Onetime Task*



The **Schedule a run-once task** window appears. Enter the description of the request in the **Requests** field. Click the arrow in the **Month**, **Day**, **Hour**, and **Minute** fields to get a list of options. When you select the Month and Day, the current Year is filled in automatically. If you select a Month and Day that is already past, the Year is set to the next year. Past requests can be retrieved and edited with the **Requests** pull-down.

The screenshot shows a Windows-style dialog box titled "Schedule a run-once task". It features a "Requests" list box at the top, which is currently empty. Below this are two sections: "Set Date" and "Set Time". The "Set Date" section contains three fields: "Year" (a text box), "Month" (a dropdown menu), and "Day" (a dropdown menu). The "Set Time" section contains two fields: "Hour" (a dropdown menu) and "Minute" (a dropdown menu). At the bottom of the dialog are two buttons: "OK" and "Cancel".

After you have made selections for all of the entries, your window would look like the following window. In this case a task is scheduled for May 31 at 12:00 PM.

**Schedule a run-once task**

Requests  
Transfer FTF.INI to E:\Temp

Set Date  
Year: 2000  
Month: May  
Day: 31

Set Time  
Hour: 12 PM  
Minute: 00

Current schedule : 2000:05:31:12:00

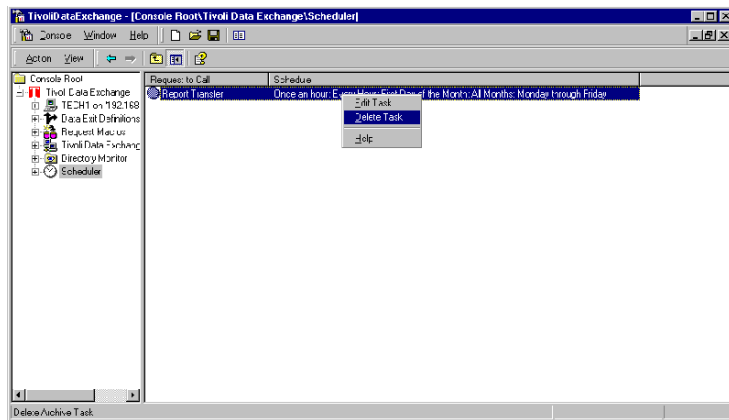
OK Cancel

The scheduled task now appears in the task display for the Scheduler.

Request to Call	Schedule
Transfer FTF.INI to E:\Temp	Minutes 0,1,2,3,4,5,10,15; Hours 10,14,16; Days 7,14; All Months; Monday through Friday
Transfer FTF.INI to E:\Temp	5/31/2000 12:00 PM

## Deleting a Scheduled Task

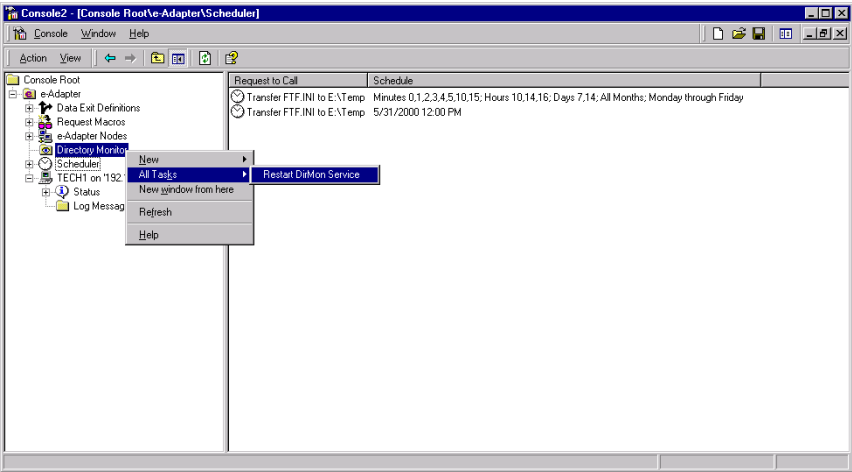
To delete a scheduled task, highlight a task in the Scheduler display and right-click the entry. Select the **All Tasks>Delete Task** option and left-click. The highlighted task is deleted.



## Restarting the Scheduler

You can manually restart the Scheduler by restarting the Directory Monitor (DirMon). Right-click **Directory Monitor** and select **All Tasks>Restart DirMon Service**.

**Tivoli Data Exchange Explorer**  
*Restarting the Scheduler*



# Directory Monitor (FTFDIRMN)

This chapter covers the Tivoli Data Exchange (TDE) Directory Monitor for both UNIX and Win 32 platforms and contains the following sections:

Section	Page
Overview	306
Configuring FTFDIRMN	306
FTFDIRMN Command-Line Arguments	310

## Assumptions

This chapter makes the following assumptions:

- TDE and the correct version of MQSeries are installed on your system. (See the *Tivoli Data Exchange Installation Guide* for MQSeries version information.)
- You understand how queues are used in TDE (FTF) and MQSeries.
- For the Windows platform, Win 32 is installed on your system. Service Pack 3 or higher needs to be installed for Win 32.

## Directory Monitor (FTFDIRMN)

### Overview

- For the UNIX platform, one of the following UNIX versions needs to be running on your system:

AIX 4.2 or higher

HP 10.20

HP 11.00

Sun 2.5.1 or higher

DEC UNIX 4.0 or higher

SCO 5.05

- For the Linux platform, kernel 2.2.16 is supported.

Both server and client are supported on Win 32, UNIX, and Linux platforms, where applicable.

## Overview

FTFDIRMN provides a way for TDE FTF requests to be submitted automatically, without manual intervention. FTFDIRMN polls a specified directory for the arrival of files. When a file arrives that meets the specified criteria, then an TDE FTF request is submitted for the file.

An TDE FTF transfer occurs when a new file is created or when a file in the monitored directory is modified. This takes care of cases where the same filename is used repeatedly to feed TDE FTF.

## Configuring FTFDIRMN

FTFDIRMN uses a rules file set up in the `ftfdirmn.ini` file that contains startup configuration settings to allow transactions to be executed.

FTFDIRMN also uses two stanzas in the `ftfconfig.ini` file.

## FTFDIRMN Rules File

The rules file can have multiple instances of FTFDIRMN. Multiple rules can be applied to a single instance of FTFDIRMN; for example, to provide different transfer arguments for different file types. The rules file must contain the following required attributes used by FTFDIRMN to execute transactions.



- **DirectoryMonitored** – The directory to be monitored.
- **PollInterval** – The time interval, in seconds, to poll the directory for files.
- **SyncQueue** – The synchronization queue used for tracking transmitted files.

---

**Note:**

The SyncQueue must be unique for each instance of FTFDIRMN.

---

- **NumFilePatterns** – The number of file types to monitor.
- **FilePattern.n** – The type of file to look for in the specified directory.
- **TransferOptions.n** – The FTF command-line options for file transfer. The -spath and -dpath options should be omitted.
- **TargetPlatform.n** – The destination platform of the file type specified.  
**Valid values:** Win 32, 4690, UNIX, MVSPS, MVSPDS, MVSGDG, AS400, AS400MEM, VMS
- **TargetDirectory.n** – The destination directory of the file type specified.
- **TargetFormat.n** – The target filename format. **Valid values:** “0” through “9”, “.” and “\*”

## Directory Monitor (FTFDIRMN)

### Configuring FTFDIRMN

The following table shows the -dpath value built according to the Target entries, based on an -spath value of [abcdefghij.txt]:

TargetPlatform	TargetDirectory	TargetFormat	Full Target Filename
Win 32, 4690	C:\DEST	8.3	C:\DEST\Abcdefgh.txt
Win 32	C:\DEST	*.2	C:\DEST\Abcdefghij.tx
Win 32, 4690	C:\DEST	3.*	C:\DEST\Abc.txt
Win 32, 4690	C:\DEST	6	C:\DEST\Abcdef
Win 32	C:\DEST	*	C:\DEST\Abcdefghij
Win 32	C:\DEST	0	C:\DEST
UNIX	/dest	8.3	/dest/abcdefgh.txt
UNIX	/dest	*.2	/dest/abcdefghij.tx
UNIX	/dest	3.*	/dest/abc.txt
UNIX	/dest	6	/dest/abcdef
UNIX	/dest	*	/dest/abcdefghij
UNIX	/dest	0	/dest
MVSPS	A.DEST	8.3	A.DEST.abcdefgh.txt
MVSPS	A.DEST	3.*	A.DEST.abc.txt
MVSPS	A.DEST	6	A.DEST.abcdef
MVSPS	A.DEST	0	A.DEST
MVSPDS	A.DEST	6	A.DEST(abcdef)
MVSPDS	A.DEST(MEMBER)	0	A.DEST(MEMBER)
MVSGDG	A.DEST	8.3	A.DEST(+1)
MVSGDG	A.DEST	*.2	A.DEST(+1)
MVSGDG	A.DEST	3.*	A.DEST(+1)
MVSGDG	A.DEST	6	A.DEST(+1)
MVSGDG	A.DEST	*	A.DEST(+1)
AS400	DEST	6	DEST/abcdef
AS400MEM	DEST	6	DEST(abcdef)
AS400MEM	DEST/FILENAME	6	DEST/FILENAME (abcdef)
VMS	DEVICE:[DEST]	8.3	DEVICE:[DEST]abcdefgh.txt
VMS	DEVICE:[DEST]	*.2	DEVICE:[DEST]abcdefghij.tx
VMS	DEVICE:[DEST]	3.*	DEVICE:[DEST]abc.txt

TargetPlatform	TargetDirectory	TargetFormat	Full Target Filename
VMS	DEVICE:[DEST]	6	DEVICE:[DEST]abcdef
VMS	DEVICE:[DEST]	*	DEVICE:[DEST]abcdefghij
VMS	DEVICE:[DEST]FILE	0	DEVICE:[DEST]FILE

It is the user's responsibility to know what value of TargetFormat produces a valid or invalid target filename for the platform specified.

## Sample Rules File

The following sample `ftfdirmn.ini` file sets up two instances of FTFDIRMN. The first instance monitors the Source directory on C drive for text, log, and configuration files. The second instance monitors the Source directory on D drive for all file types.

```
FTFDirmn:
    name = Instance.1, Instance.2

Instance.1:
    DirectoryMonitored=C:\SOURCE
    PollInterval=600
    SyncQueue=FTFDIRMN.SYNC
    NumFilePatterns=3

    FilePattern.1=*.TXT
    TransferOptions.1=-lqm QMGR1 -sqm QMGR1 -dqm QMGR2
    TargetPlatform.1=NT
    TargetDirectory.1=C:\DEST
    TargetFormat.1=*. *

    FilePattern.2=*.LOG
    TransferOptions.2=-lqm QMGR1 -sqm QMGR1 -dqm QMGR2
    TargetPlatform.2=NT
    TargetDirectory.2=C:\DEST
    TargetFormat.2=*. *

    FilePattern.3=*.INI
    TransferOptions.3=-lqm QMGR1 -sqm QMGR1 -dqm QMGR2
    TargetPlatform.3=NT
    TargetDirectory.3=D:\DEST
    TargetFormat.3=*. *
```

## Directory Monitor (FTFDIRMN)

### *FTFDIRMN Command-Line Arguments*

```
Instance.2:
DirectoryMonitored=D:\SOURCE
PollInterval=600
SyncQueue=FTFDIRMN.SYNC.1
NumFilePatterns=1
FilePattern.1=*. *
TransferOptions.1=-lqm QMGR1 -sqm QMGR1 -dqm QMGR2 -delsrc
TargetPlatform.1=NT
TargetDirectory.1=D:\DEST
TargetFormat.1=*. *
```

## FTFDIRMN Configuration File

The two stanzas in the `ftfconfig.ini` file that relate to FTFDIRMN are as follows:

- **DirmonNumInstances** – The number of instances for FTFDIRMN.
- **DirmonControlQueue** – Name of the queue FTFDIRMN will monitor for shutdown messages.

### Sample Configuration File

The following is a sample of the FTFDIRMN configuration file:

```
DirmonNumInstances=1
DirmonControlQueue=FTFDIRMN.CONTROL
```

## FTFDIRMN Command-Line Arguments

The FTFDIRMN command line accepts the following arguments:

```
FTFDIRMN -lqm localQueueMgr -nodename nodeName
[-cfile configFile | -cq configQueueName]
-ofile optionsFile -lfile logFile -version
-rfile rulesFile -instance instanceNumber
```

---

### Note:

**-rfile** and **-instance** are required options.

---

Where:

- **-lqm** *localQueueMgr* – Name of the queue manager to which the FTFDIRMN command will attach. If the lqm is not specified, the FTFDIRMN command attempts to connect to the default queue manager. Otherwise, it tries to connect to the local queue manager (lqm) specified.
- **-nodename** *nodeName* – Associates the node name with the instance of TDE FTF directory monitor. The nodeName must be defined in the configuration file as FTFNodeAlias or FTFNodeOverride.
- **-cfile** *configFile* – Contains the fully qualified path and filename for the TDE FTF configuration file. If you specify both a -cfile and a -cq argument in the same command, the -cq argument takes precedence.
- **-cq** *configQueueName* – Specifies the queue from which the configuration information is to be retrieved for this TDE FTF instance. The -cq argument points TDE FTF to the queue name rather than to the standard configuration file. If you specify both a -cfile and a -cq argument, the -cq argument takes precedence.

### **Configuration File and Queue Order of Precedence**

If you do not specify either a configuration file or a configuration queue, TDE FTF checks the FTF\_CONFIG\_QUEUE environment variable and uses the specified queue. If this environment variable is not set, TDE FTF checks the FTF\_CONFIG\_FILE environment variable and uses the specified file. If neither environment variable is set and no command-line argument is set, the command fails.

- **-ofile** *optionsFile* – Contains the fully qualified path and filename of a text file used to contain command-line arguments for the FTFDIRMN command. In the options file, you can set any of the command-line arguments that can be set for the FTFDIRMN command. Any values specified on the command line override the values in the options file.

## Directory Monitor (FTFDIRMN)

### *FTFDIRMN Command-Line Arguments*

- **-lfile** *logFile* – Contains the log file to which the TDE FTF directory monitor writes.

---

#### **Note:**

On UNIX platforms, if you start an TDE FTF component as a background process, log information is still generated to standard output. To eliminate standard output, redirect it to /dev/null.

---

- **-version** – Returns the current TDE FTF version, release, and patch number. Any command entered with this argument ignores all other arguments and returns the version information.
- **-rfile** *rulesFile* – The text file containing startup configuration settings for FTFDIRMN, which allows FTFDIRMN to execute transactions. This is a required argument. Multiple rules can be applied to a single instance of FTFDIRMN: for example, to provide different transfer arguments for different file types.
- **-instance** *instanceNumber* – The number of the FTFDIRMN instance. This is a required argument.

# **Tivoli Data Exchange Client for 4690**

This chapter describes how to install and configure Tivoli Data Exchange (TDE) Client on the 4690 platform. The chapter also describes the commands which start up the TDE Client components within the 4690 environment. Once the TDE Manager, Sender, and Receiver components are configured, the data transfers may be initiated. The chapter includes the following sections:

<b>Section</b>	<b>Page</b>
Overview	314
System and Installation Requirements	316
Installing the Client for 4690	316
Initializing the Client for 4690	318
Configuring the Client for 4690	321
Setting Up Clients on Another Platform	326
Using Exits on the 4690 Client	329
Tivoli Data Exchange Commands	329

## Assumptions

This chapter makes the following assumptions:

- You have appropriate permission to install software on the installation machine.
- You have a working knowledge of TDE Client components and their interrelationships.
- You know how to access and use a command line in the appropriate operating system.
- You have connectivity to a TDE server instance.

## Overview

This sections explains the features and benefits of MQSeries Client architecture.

## What Is an MQSeries Client?

---

### **Note:**

The MQSeries Client functionality is embedded within the TDE Client for 4690. Therefore, installation of the MQSeries Client is not required.

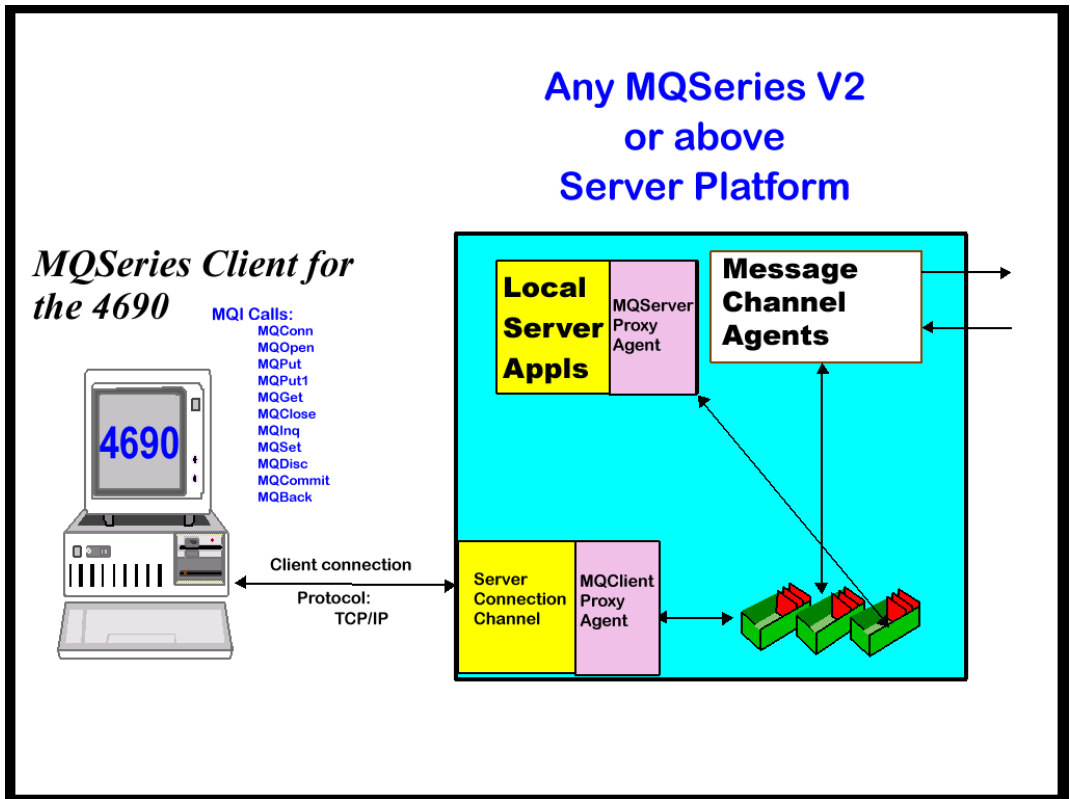
---

An MQSeries client is part of the MQSeries product and can be installed on a machine without installing the full queue manager. It accepts Message Queue Interface (MQI) calls from application programs and passes these MQI requests to an MQSeries server.

The MQSeries server is a full queue manager that can accept MQI calls directly from application programs that are running on the server processor. In addition, it accepts MQI requests from MQSeries clients.

The following configuration has the MQSeries client running on one machine (the client machine) and the queue manager running on a different machine.





## Why Use MQSeries Clients?

Using MQSeries clients gives you an efficient way to implement MQSeries messaging and queuing without the footprint of a server installation. You can run the MQSeries client on one machine and the MQSeries server on a different machine, either physically or virtually. The benefits are:

- There is no need for a full MQSeries implementation on the client machine. This allows for installation on such platforms as: DOS, Windows 3.1, Windows 95, or 4690 POS Controller.
- Hardware requirements on the client system are reduced.
- System administration requirements are reduced.

## **Tivoli Data Exchange Client for 4690**

### *System and Installation Requirements*

- An MQSeries application running on a client can connect to multiple queue managers on different systems.
- Alternative channels using different transmission protocols may be used.

For more information about MQSeries Clients, see the following IBM supplied manual:

<b>Title</b>	<b>Number</b>
MQSeries Clients	GC33-1632-03

## **System and Installation Requirements**

The following are the system requirements for the successful installation and operation of the TDE Client for 4690:

- 4690 Store System
- IBM 4690 OS/386 V1 or V2
- TCP/IP Protocol support

The following conditions must exist before you can install TDE Client for 4690:

- You must have the proper authority to install TDE Client for 4690 in the desired directories.
- An MQSeries server environment must be accessible.

No group or user is required for the installation. However, at run time, TDE Client for 4690 uses MQSeries, which may require access to the *mqm* group.

## **Installing the Client for 4690**

To install TDE Client for 4690 locally from a floppy diskette:

1. Insert the diskette into the 4690 floppy drive.
2. Create a directory on the C drive to install into (example: md eAdpCLNT).
3. Copy the contents of the floppy into the new directory.

To install TDE Client for 4690 from a CD-ROM on another machine:

1. Put the CD-ROM into the drive that has access to the 4690.
2. CD (change directory) to the 4690\Client directory.
3. Create the target directory on the 4690.
4. FTP all files in binary from the 4690\Client directory to the 4690 target directory.

## **Adding Logical Filenames**

- FTFCFGFI – must be set to the name of the TDE Client for 4690 configuration file with the fully qualified path.

Example:

Define FTFCFGFI = C:\FTFMQ\CONFIG\FTFCONFI.INI

- FTFCFGQ – can optionally be set to specify the default configuration queue.

Example:

Define FTFCFGQ = FTFCFG

To permanently define these logical filenames as part of the system environment, follow the same procedure as defined on page 322 for MQSERVER.

## **Specifying File Distribution Types on the Command Line**

There are five distribution types of files on the 4690. They are associated with the target type (-ttype) and log file type (-ltype) arguments. Valid file distribution values may be 1 to 5, as follows:

- **1** – Local. Only the local machine is involved in the process. This is the default value. If the type is not specified, then it is set to **1**.

## Tivoli Data Exchange Client for 4690

### *Initializing the Client for 4690*

- **2** – Mirrored on update. In a dual configuration the file is also written to the second 4690 controller; with Type 2 distribution type, the file is mirrored on a record-by-record basis.
- **3** – Mirrored on close. In a dual configuration the file is also written to the second 4690 controller; with Type 3 distribution type, the file is written to the second controller after the full file is written and closed on the first controller.
- **4** – Compound on update. Use when you have multiple 4690 controllers in the configuration. The file is written to all controllers on a record-by-record basis. This works like Type 2 if you have only two controllers in the configuration.
- **5** – Compound on close. Use when you have multiple 4690 controllers in the configuration. The file is written to all controllers after the full file is written and closed. This works like Type 3 if you have only two controllers in the configuration.

## Specifying the Target File Distribution Type in the Configuration

You can also set up the TDE Receiver to specify the distribution type for the target files, instead of adding the `-ttype` argument to the command line. To do this, use the `4690TargetType` stanza in the `ftfconfig.ini` file with the appropriate target type value, as follows:

```
4690TargetType=2
```

## Initializing the Client for 4690

This section describes how to start up the TDE Client for 4690 components in a 4690 environment using the command line.

### Tivoli Data Exchange Manager Startup

To start the TDE Manager from the command line, enter the following command:

```
ftfmgr -lqm localQMgr -cfile configFile -lfile  
logFile -ltype logType -nodename nodeName
```

Where:

- **-lqm** *localQMgr* – Name of the MQSeries queue manager to which you want to connect.
- **-cfile** *configFile* – The fully qualified path and filename for the TDE Client for 4690 configuration file. If this value is not set, it defaults to the value specified in the FTFCFGFI environment variable. For more information about this logical filename, See “Installing the Client for 4690” on page -316..
- **-lfile** *logFile* – The fully qualified path and filename for the TDE Manager log file. If you do not specify a log file, output is sent to the monitor.
- **-ltype** *logType* – The distribution type of the log file. Applies to the 4690 Client only. See “Specifying File Distribution Types on the Command Line” for more information. **Valid values:** 1 - 5 **Default value:** 1
- **-nodename** *nodeName* – Associates a client definition that is described in the alias or override section of the configuration file with the message that is being sent. The nodeName can be anything, but it must be defined in the FTF configuration table at every node that wishes to participate with it.

## **Tivoli Data Exchange Sender Startup**

To start the TDE Sender, enter the following command:

```
ftfsdr -lqm localQMgr -cfile configFile -lfile  
logFile -ltype logType -nodename nodeName
```

Where:

- **-lqm** *localQMgr* – Name of the MQSeries queue manager to which you want to connect.
- **-cfile** *configFile* – The fully qualified path and filename for the TDE Client for 4690 configuration file. If this value is not set, it defaults to the value specified in the FTFCFGFI environment variable. For more information about this logical filename, See “Installing the Client for 4690” on page -316..

## Tivoli Data Exchange Client for 4690

### Initializing the Client for 4690

- **-lfile** *logFile* – The fully qualified path and filename for the TDE Sender log file. If you do not specify a log file, output is sent to the monitor.
- **-ltype** *logType* – The distribution type of the log file. Applies to the 4690 Client only. See “Specifying File Distribution Types on the Command Line” for more information. **Valid values:** 1 - 5 **Default value:** 1
- **-nodename** *nodeName* – Associates a client definition that is described in the alias or override section of the configuration file with the message that is being sent. The nodeName can be anything, but it must be defined in the FTF configuration table at every node that wishes to participate with it.

## Tivoli Data Exchange Receiver Startup

To start the TDE Receiver, enter the following command:

```
ftfrcv -lqm localQMGr -cfile configFile  
      -lfile logFile -ltype logType -nodename nodeName  
      -ttype targetType
```

Where:

- **-lqm** *localQMGr* – Name of the MQSeries queue manager to which you want to connect.
- **-cfile** *configFile* – The fully qualified path and filename for the TDE Client for 4690 configuration file. If this value is not set, it defaults to the value specified in the FTFCFGFI environment variable. For more information about this environment variable, See “Installing the Client for 4690” on page -316..
- **-lfile** *logFile* – The fully qualified path and filename for the TDE Receiver log file. If you do not specify a log file, output is sent to the monitor.
- **-ltype** *logType* – The distribution type of the log file. Applies to the 4690 Client only. See “Specifying File Distribution Types on the Command Line” for more information. **Valid values:** 1 - 5 **Default value:** 1

- **-nodename** *nodeName* – Associates a client definition that is described in the alias or override section of the configuration file with the message that is being sent. The nodeName can be anything, but it must be defined in the FTF configuration table at every node that wishes to participate with it.
- **-ttype** *targetType* – The distribution type for the target files. Applies to the 4690 Client only. See “Specifying File Distribution Types on the Command Line” for more information. **Valid values:** 1 - 5 **Default value:** 1

## Configuring the Client for 4690

The following tables illustrate how to configure the MQSeries Client as a background process, how to set up the background processes for the TDE components (Manager, Sender, and Receiver), and how to start the components in the background.

### Setting Up the MQSeries Client

To connect the 4690 Client to a server, the MQSERVER variable must be defined. Follow the steps below from the SYSTEM MAIN MENU (displayed at startup or by pressing <Alt> + <PrtScn/SysRq> then selecting option **s** **Start new application**):

Screen	Action	Result
SYSTEM MAIN MENU	Select option <b>4 Installation and Update Aids</b>	INSTALLATION AND UPDATE AIDS screen appears.
INSTALLATION AND UPDATE AIDS	Select option <b>1 Change Configuration Data</b>	CONFIGURATION screen appears.
CONFIGURATION	Select option <b>2 Controller Configuration</b>	LAN CONFIGURATION screen appears.
LAN CONFIGURATION	Press <Enter>	CONTROLLER CONFIGURATION screen appears.

## Tivoli Data Exchange Client for 4690

### Configuring the Client for 4690

CONTROLLER CONFIGURATION	Press <Enter>	CONTROLLER CONFIGURATION STORE CONTROLLER screen appears.
CONTROLLER CONFIGURATION STORE CONTROLLER	Press <Tab> until the cursor is on the line <b>User Logical File Names</b> . Place an <b>X</b> in front of the line and press <Enter>.	USER LOGICAL FILE NAMES STORE CONTROLLER screen appears.
USER LOGICAL FILE NAMES STORE CONTROLLER	Press <Enter>	Cursor is on the line Type the logical filename being processed.
USER LOGICAL FILE NAMES STORE CONTROLLER	Type <b>MQSERVER</b> and press <Enter>	DEFINE LOGICAL FILE NAMES STORE CONTROLLER screen appears.
DEFINE LOGICAL FILE NAMES STORE CONTROLLER	Type <b>QMGRNAME/TCP/XXX. XXX.XXX.XXX(NNNN)</b> where <b>QMGRNAME</b> is the name of the MQSeries Server queue manager that the TDE client will connect to, <b>XXX.XXX.XXX.XXX</b> is the IP address of the server, and <b>NNNN</b> is the port number of the server (not required if default port 1414). Press <Enter>.	USER LOGICAL FILE NAMES STORE CONTROLLER screen appears.
USER LOGICAL FILE NAMES STORE CONTROLLER	Select option <b>4 Display Existing Logical File Names</b>	The newly defined MQSERVER variable is displayed.
DISPLAY USER LOGICAL FILE NAMES STORE CONTROLLER	Press <F3> three times	CONFIGURATION screen appears.
CONFIGURATION	Select option <b>4 Activate Configuration</b>	ACTIVATE CONFIGURATION screen appears.



ACTIVATE CONFIGURATION	Select option <b>2 Controller Con- figuration</b>	The message Configuration changes are being verified will appear for a few minutes. When completed the message Additional messages available. Press F10 will appear.
ACTIVATE CONFIGURATION	Press <F10>	MESSAGE screen appears. Verify there are no error messages.
MESSAGE	Press <F10>	ACTIVATE CONFIGURATION screen appears.
ACTIVATE CONFIGURATION	Press <F3> until SYSTEM MAIN MENU appears	Completed.

## Setting Up Background Processes for Components

To setup the TDE components (Manager, Sender, and Receiver) to execute in the background, follow the steps below from the SYSTEM MAIN MENU (displayed at startup or by pressing <Alt> + <PrtScn/SysRq> then selecting option **s Start new application**):

Screen/Description	Action	Result
SYSTEM MAIN MENU	Select option <b>4 Installation and Update Aids</b>	INSTALLATION AND UPDATE AIDS screen appears.
INSTALLATION AND UPDATE AIDS	Select option <b>1 Change Configuration Data</b>	CONFIGURATION screen appears.
CONFIGURATION	Select option <b>2 Controller Configuration</b>	LAN CONFIGURATION screen appears.
LAN CONFIGURATION	Press <Enter>	CONTROLLER CONFIGURATION screen appears.
CONTROLLER CONFIGURATION	Press <Enter>	CONTROLLER CONFIGURATION STORE CONTROLLER screen appears.

## Tivoli Data Exchange Client for 4690

### Configuring the Client for 4690

CONTROLLER CONFIGURATION STORE CONTROLLER	Press <Tab> until the cursor is on the line <b>Background Application</b> . Place an X in front of the line and press <Enter>.	BACKGROUND APPLICATION STORE CONTROLLER screen appears.
<b>Repeat the following steps for each component (ftfmgr, ftfsdr, and ftfrcv)</b>		
BACKGROUND APPLICATION STORE CONTROLLER	Select option <b>1 Define a Background Application</b>	DEFINE BACKGROUND APPLICATION STORE CONTROLLER screen appears.
DEFINE BACKGROUND APPLICATION STORE CONTROLLER	In the <b>INITIAL MESSAGE</b> field, type a meaningful description for the component, such as <b>FTFMGR</b> . Press <Tab>.	Cursor is in the PROGRAM NAME field.
DEFINE BACKGROUND APPLICATION STORE CONTROLLER	In the <b>PROGRAM NAME</b> field, type a component executable, such as <b>"c:\ftfmq\ftfmgr.286"</b> . Assuming ftfmq is the directory where TDE has been installed. The valid executable names are ftfmgr.286, ftfsdr.286, and ftfrcv.286. Press <Tab>.	Cursor is in the PARAMETER LIST field.
DEFINE BACKGROUND APPLICATION STORE CONTROLLER	In the PARAMETER LIST field, type the ofile filename (to be created later), such as <b>"-ofile c:\ftfmq\mgrofile.txt"</b> . Press <Page Down>	Cursor is in the IPL START field.
DEFINE BACKGROUND APPLICATION STORE CONTROLLER	If you want the component to be started at IPL, then type <b>Y</b> ; otherwise, type <b>N</b> ; press <Enter>	BACKGROUND APPLICATION STORE CONTROLLER screen appears.
<b>End of Repetition</b>		

BACKGROUND APPLICATION STORE CONTROLLER	Press <F3> until <b>CONFIGURATION</b> screen appears	CONFIGURATION screen appears.
CONFIGURATION	Select option <b>4 Activate Configuration</b>	ACTIVATE CONFIGURATION screen appears.
ACTIVATE CONFIGURATION	Select option <b>2 Controller Con- figuration</b>	The message Configuration changes are being verified will appear for a few minutes. When completed the message Additional messages available. Press F10 will appear.
ACTIVATE CONFIGURATION	Press <F10>	MESSAGE screen appears. Verify there are no error messages.
MESSAGE	Press <F10>	ACTIVATE CONFIGURATION screen appears.
ACTIVATE CONFIGURATION	Press <F3> until <b>SYSTEM MAIN MENU</b> screen appears	SYSTEM MAIN MENU screen appears.
Create the ofile.	Create the <b>mgrofile.txt</b> with the following line: <b>-lqm QMGR -nodename NODE -cfile c:\ftfmq\ftfconfi.ini -lfile c:\ftfmq\ftfmgr.log</b> where <i>QMGR</i> is the Queue Manager name on the server and <i>NODE</i> is the name of the 4690 Client.	File has been created. Create an ofile for the other components if needed.
IPL machine.	IPL machine	Machine is IPL'd.

## Starting Background Processes for Components

To start the TDE components (Manager, Sender, and Receiver) created in the previous section, follow the steps below from the BACKGROUND APPLICATION CONTROL (displayed by pressing <Alt> + <PrtScn/SysRq> then selecting option **b Access the Background Application Control screen**):

Screen	Action	Result
BACKGROUND APPLICATION CONTROL	Press <Tab> to the background process for the component you want to start. You may have to press <Page Down> if you do not see your process on the current screen. Press <F7> to start the process.	STATUS=ACTIVE will be displayed. Repeat step for each desired component.
BACKGROUND APPLICATION CONTROL	Press <F3>	Completed.

## Setting Up Clients on Another Platform

The startup parameters for the TDE Sender, Manager, and Receiver can be set up for client deliveries only by using the -nodename argument. The -nodename argument accepts an identifier that will be used in the respective source queue manager (-sqm) or destination queue manager (-dqm) arguments of the FTF request. For example, if the FTF request looks as follows when using MQSeries servers:

```
FTF -sqm MQM1 -dqm MQM2 -spath C:\FILE.1  
-dpath C:\FILE.2
```

Then the command when working with clients would look like the following:

```
FTF -sqm CLIENTA -dqm CLIENTB -spath C:\FILE.1  
-dpath C:\FILE.2
```

However, the TDE components on CLIENTA and CLIENTB need to be started with the -nodename CLIENTA and -nodename CLIENTB, respectively. This name must be specified in the FTFNodeAlias section, or defined in the FTFNodeOverride section of the configuration file at every node that wishes to participate with this client.

The client queues must be defined on the server queue manager. The queues used by the FTFNodeAlias section use the Alias's name and the default section of the configuration file to define the queues. For example, a queue named CLIENTA.FTFMGR.CONTROL must be defined on queue manager MQM1. All of the working queues must be defined in this manner.

See the “FTFNodeAlias Section” in the Configuration chapter of the *Tivoli Data Exchange Installation Guide* and the “Additional Information” section for “FTFMGR” in the *Tivoli Data Exchange Technical Reference* for more information on FTFNodeAlias. The entry in the FTFNodeAlias section of the configuration file would look as follows:

```
FTFNodeAlias:  
AliasQueueManager=MQM1  
Aliases=CLIENTA, CLIENTB
```

## **Tivoli Data Exchange Client for 4690**

### *Setting Up Clients on Another Platform*

All of the client queues specified in the FTFNodeOverride section must be defined on the queue manager specified by the Queue Manager stanza. For example, all of the A.\* queues need to be defined on the MQM1 queue manager. The entry in the FTFNodeOverride section of the configuration file would look as follows:

```
FTFNodeOverride:
name=CLIENTA, CLIENTB

CLIENTA:
QueueManager=MQM1
ManagerControlQueue=A.FTFMGR.CONTROL
ManagerSyncQueue=A.FTFMGR.SYNC

SenderNumInstances=1
SenderStageControlQueue=A.FTFSDR.STAGE.CONTROL
SenderControlQueue=A.FTFSDR.CONTROL
SenderSyncQueue=A.FTFSDR.SYNC
SenderStageQueue=A.FTFSDR.STAGE
SenderMaxStageQueues=1
SenderSystemQueue=A.FTFSDR.SYSTEM
SenderCancel=NO

ReceiverNumInstances=1
ReceiverControlQueue=A.FTFRCV.CONTROL
ReceiverSyncQueue=A.FTFRCV.SYNC
ReceiverStageQueue=A.FTFRCV.STAGE
ReceiverSystemQueue=A.FTFRCV.SYSTEM
ReceiverCancel=NO

CLIENTB:
QueueManager=MQM1
ReceiverNumInstances=1
ReceiverControlQueue=B.FTFRCV.CONTROL
ReceiverSyncQueue=B.FTFRCV.SYNC
ReceiverStageQueue=B.FTFRCV.STAGE
ReceiverSystemQueue=B.FTFRCV.SYSTEM
ReceiverCancel=NO
```

## Using Exits on the 4690 Client

The processing of TDE user exits is slightly different on the 4690 platform.

---

### Note:

For a general overview on how to use the TDE user exits, refer to the “User Exits” chapter in the *Tivoli Data Exchange User's Guide*.

---

All of the user exit points are supported on the 4690 but instead of invoking a dynamic load library (DLL), you use the -exitdata option to submit a command-line request. You submit the command-line request in a similar manner to that of the Tivoli Data Exchange sample exits on other platforms. You can invoke system commands, user-written batch files and executables.

The following sample request invokes the Manager Post-Process exit (4) to move the source file to an archive directory. Since the Manager process being used in this transaction is running on the same machine as the source file, it can access the source file via its Post-Process exit. Because there are no DLLs involved in the exit processing on the 4690, TDE requires that you enter the text “4690” for both the -exitdll and -exitentry options on the request. The sample request is as follows:

```
FTF -lqm UNIXQMGR -oqm 4690QMGR -sqm 4690QMGR  
-spath C:/store1/prices.txt -dqm UNIXQMGR  
-dpath /region2/store1/prices.txt -exit 4  
-exitdll 4690 -exitentry 4690 -exitdata "move  
C:/store1/prices.txt C:/archive/store1.prices.txt"
```

## Tivoli Data Exchange Commands

### Overview

The commands in this section allow you to start specified TDE Client for 4690 components or the entire TDE Client for 4690 environment. These commands can be used from any command line on the specified operating systems.

## **Tivoli Data Exchange Client for 4690**

### *Tivoli Data Exchange Commands*

You can use the following commands to manipulate TDE Client for 4690 components.

#### **FTF**

The FTF command allows you to perform data transfers from one TDE component to another. You can find an extensive explanation of this command in the “Interface Commands” chapter of the *Tivoli Data Exchange Technical Reference*.

#### **FTFMGR**

The FTFMGR command allows you to start a TDE Manager according to the conditions set in the command-line parameters. You can find an extensive explanation of this command in the “Component Configuration Commands” chapter of the *Tivoli Data Exchange Technical Reference*.

#### **FTFRCV**

The FTFRCV command allows you to start a TDE Receiver. You can find an extensive explanation of this command in the “Component Configuration Commands” chapter of the *Tivoli Data Exchange Technical Reference*.

#### **FTFSDR**

The FTFSDR command allows you to start a TDE Sender. You can find an extensive explanation of this command in the “Component Configuration Commands” chapter of the *Tivoli Data Exchange Technical Reference*.

#### **FTFPING**

FTFPING command allows you to send a test data transfer to TDE components that are currently running. You can find an extensive explanation of this command in the *Tivoli Data Exchange Technical Reference* manual in the “Tivoli Data Exchange Interface Commands” chapter.



## **FTFEND**

The FTFEND command allows you to stop TDE components that are currently running. You can find an extensive explanation of this command in the “Component Configuration Commands” chapter of the *Tivoli Data Exchange Technical Reference*.



# Using Pools

This chapter describes how to use pools with Tivoli Data Exchange (TDE) to improve data-transfer performance and to control the queues used during a data transfer. A case study is presented to illustrate the tasks required to configure and use pools.

This chapter contains the following sections:

<b>Section</b>	<b>Page</b>
About Pools	334
Using Pools with FTF	335
Designing the Pools	335
Implementing the MQSeries Objects	338
Specifying a Pool	344

## Assumptions

This chapter makes the following assumptions:

- You understand the roles of the TDE FTF Manager, FTF Sender, and FTF Receiver in the execution of a data-transfer request.
- You understand the C programming language.

## About Pools

FTF pools allow you to create and use logical queues made up of multiple physical queues. Pools reside on the FTF Receiver and allow you to increase throughput, segregate data-transfer traffic, and overcome MQSeries queue capacity limitations.

You can have multiple pools with one set of channels using the default Sender and Receiver nodes, and vice versa. The pool definitions must be the same on both the sending and receiving platforms, as well as in the FTF configuration file.

## Increasing Throughput

Pools increase throughput by using multiple physical connections between the FTF Sender and the FTF Receiver to deliver the messages that make up a file. For instance, if four connections exist between the FTF Sender and the FTF Receiver, you can use channels to send messages over all four connections, rather than only one of them.

## Load Balancing

Pools allow you to break up the traffic created across channels for a single data-transfer request. You can specify the number of messages to be sent across each channel in a data-transfer request before the next channel is used. If you identify the appropriate number of messages, you can spread the data-transfer requests equally across all channels participating in the data transfer.

## Overcoming Queue Capacity Limitations

Pools allow you to effectively increase the MQSeries queue capacity limitations. MQSeries version 2.x has a queue capacity of 320 megabytes (MB). MQSeries version 5.x has a queue capacity of 2 gigabytes (GB). Pools allow you to spread out the traffic in a data transfer, essentially increasing your queue capacity by about 100 percent with each queue contained in the pool.

## Using Pools with FTF

To use pools with FTF, you must perform the following tasks:

1. Design the pool scheme.
2. Implement the MQSeries objects (see page 338).
3. Define the pools in the FTF configuration file (see page 341).
4. Set the appropriate pool values in the FTF interface (see page 344).

## Designing the Pools

This first step in using, configuring, and implementing queue pools is determining the structure of the pools you are implementing. Pool design is application specific; the design you use is dictated entirely by your processing requirements and constraints.

## Case Study

This chapter uses a case study to illustrate the steps required for setting up FTF pools. In the case study, pools are used to assist with capacity planning and scalability concerns.

In this instance, the primary purpose of using pools is to overcome MQSeries queue-capacity limits and to prevent bottlenecks during large data transfers. Pools are also used in this scenario to allow you to use four physical connections during a transfer, which both increases speed and reduces the possibility of a bottleneck.

The following environment exists for this case study:

- Four T1 connections exist between the FTF Sender and the FTF Receiver.
- The files being transferred are usually about 2GB in size and the information in them is timely. These files are transferred from the FTF Sender and the FTF Receiver more often than any other files.
- The number of transfers throughout the day is moderate.

## Using Pools

### *Designing the Pools*

In this case, you would perform the following tasks to determine the best design for using FTF pools:

1. Determine the number of required queues.
2. Determine the number of required queue manager aliases.
3. Determine the optimal number of channels.

### Determining the Number of Queues

To begin the design, you must determine the number of queues required to effectively perform the data transfer. Two types of queues must be created for the pools to work:

- Transmission queues on the FTF Sender. These queues are used to send messages to the pool queues on the FTF Receiver.
- Local queues on the FTF Receiver. FTF uses these queues to create the pools.

To determine the number of queues, the following formula can provide a reasonable starting point:

$$\frac{dataSize}{queueCapacity} \cdot arrivalRate = numQueues$$

Where:

- *dataSize* is the size of the data you are transferring.
- *queueCapacity* is the MQSeries queue capacity.
- *arrivalRate* is the rate at which you expect messages to arrive.
- *numQueues* is the number of transmission queues planned for the FTF Sender and the number of local queues planned for the FTF Receiver.

---

#### Note:

The queue capacity for MQSeries version 5.x is 2GB. The queue capacity for MQSeries version 2.x is 320MB.

---

In this scenario, the file size is 2GB and the desired arrival rate—based on the capacity of the physical connections—is five messages at one time. In MQSeries version 5.x, the formula for determining the number of queues is:

$$\frac{2GB}{2GB} \cdot 5 = 5$$

In MQSeries version 2.x, the formula for determining the number of queues is:

$$\frac{1074MB}{320MB} \cdot 5 = 16.78$$

The following table lists the number of transmission and local queues required for each version of MQSeries.

<b>MQSeries Version</b>	<b>Transmission Queues</b>	<b>Local Queues</b>
Version 5.x	5	5
Version 2.x	16	16

## Determining the Number of Queue Manager Aliases

Queue manager aliases are required by FTF to facilitate queue name resolution. For more information about queue name resolution, see

[www.software.ibm.com/ts/mqseries/library/manuals99/csqa6n.htm](http://www.software.ibm.com/ts/mqseries/library/manuals99/csqa6n.htm)

FTF requires queue manager aliases for each local queue after the first one. You can determine the number of queue manager aliases by subtracting one from the number of local queues. The following table lists the number of queues and queue manager aliases required for each version of MQSeries.

<b>MQSeries Version</b>	<b>Transmission Queues</b>	<b>Local Queues</b>	<b>Queue Mgr. Aliases</b>
Version 5.x	5	5	4
Version 2.x	16	16	15

## Determining the Number of Channels

Our sample scenario specifies that four T1 connections exist between the FTF Sender and the FTF Receiver. For connections with high throughput capabilities, you can specify multiple channels for each connection, to take full advantage of the connections.

If you have high throughput connections, you should consider creating one channel per transmission queue. This arrangement prevents the transmission queues being bottlenecks. The following table lists the number of queues, aliases, and channels required to support this scenario in each MQSeries version.

<b>MQSeries Version</b>	<b>Transmission Queues</b>	<b>Local Queues</b>	<b>Queue Mgr. Aliases</b>	<b>Channels</b>
Version 5.x	5	5	4	5
Version 2.x	16	16	15	16

If your connections cannot handle this volume of traffic, you might have to scale back the number of channels and queues. Even after this arrangement is implemented, you may have to tune it to reach optimal performance.

## Implementing the MQSeries Objects

The first task in implementing pools is to define the MQSeries objects specified in the design. This section describes how to perform the following implementation steps:

- Defining the transmission queues on the FTF Sender
- Defining the local queues on the FTF Receiver
- Defining the queue manager aliases on the FTF Sender
- Defining the channels

In the case study illustrating how to set up and use pools, the following assumptions are made:

- Both MQSeries and FTF have been installed and configured as required to create the enterprise used in this chapter.
- The source queue manager (sqm) running on the FTF Sender is named QM; the destination queue manager (dqm) running on the FTF Receiver is named QM2.



The examples in this section assume that the case study environment is using MQSeries version 5.x. The only difference between the version 5.x implementation and the version 2.x implementation is the number of queues required. To perform this implementation for version 2.x increase the number of queues from 5 to 16.

## Defining the Transmission Queues

In the case study for MQSeries, version 5.x, the design specifies that five transmission queues be created. As required by MQSeries, each of these transmission queues must have a unique name within their queue manager. Although these names do not require a naming convention, you should consider using a numeric suffix to differentiate the queue names. This naming convention allows you to more easily map the transmission queues to their respective queue manager aliases and FTF Receiver local queues.

For the purposes of this case study, the transmission queues are named QM2.x, QM2.x.1, QM2.x.2, QM2.x.3, and QM2.x.4.

To define the transmission queues, you would use the following MQSeries commands:

```
DEFINE QLOCAL(QM2.x) USAGE (XMITQ)
DEFINE QLOCAL(QM2.x.1) USAGE (XMITQ)
DEFINE QLOCAL(QM2.x.2) USAGE (XMITQ)
DEFINE QLOCAL(QM2.x.3) USAGE (XMITQ)
DEFINE QLOCAL(QM2.x.4) USAGE (XMITQ)
```

## Defining the Local Queues

The local queues on the FTF Receiver receive the messages sent by the transmission queues already defined on the FTF Sender. As required by MQSeries, each of these queues must have a unique name within their queue manager.

FTF can use the dynamic queue name expansion facility to resolve queue names for the queues in a pool. This process uses a base queue name (specified in the FTF configuration file) and appends a counter to each queue created after the first one.

## Using Pools

### *Implementing the MQSeries Objects*

For example, if the base queue name for the case study is *poolQueue*, using the following queue names would allow you to take advantage of dynamic queue name expansion:

- `poolQueue`
- `poolQueue.1`
- `poolQueue.2`
- `poolQueue.3`
- `poolQueue.4`

To define these queues, use the following MQSeries commands:

```
DEFINE QLOCAL(poolQueue)
DEFINE QLOCAL(poolQueue.1)
DEFINE QLOCAL(poolQueue.2)
DEFINE QLOCAL(poolQueue.3)
DEFINE QLOCAL(poolQueue.4)
```

## Defining the Queue Manager Aliases

In order to properly use multiple channels with pools, you must set up a queue manager alias for each FTF Manager after the first. Each of these aliases resolves to the dqm running on the FTF Receiver. FTF requires these queue manager aliases to facilitate queue name resolution.

To define the queue manager aliases for our example, you would use the following MQSeries commands:

```
DEFINE QREMOTE (QM2.1) RQMNAME (QM2) XMITQ(QM2.x.1)
DEFINE QREMOTE (QM2.2) RQMNAME (QM2) XMITQ(QM2.x.2)
DEFINE QREMOTE (QM2.3) RQMNAME (QM2) XMITQ(QM2.x.3)
DEFINE QREMOTE (QM2.4) RQMNAME (QM2) XMITQ(QM2.x.4)
```

## Defining the Channels

The case study design calls for five channels to be set up, one for each transmission queue. For information about setting up MQSeries channels, see the *MQSeries Intercommunication* manual, available online from IBM.

## Defining Pools in the FTF Configuration File

Within FTF, you must define pools in the FTF configuration file's Data Pools section. This section determines the pool names, the name of the queues in the pool, and all other pool-related settings. For complete information about the FTF configuration file, see "FTF Configuration" in the *Tivoli Data Exchange Installation Guide*.

Specify the following settings in the Data Pools section:

- **Pool name** – Names of the queue pools set up with FTF. One name must appear for each queue pool.
- **QueueName** – Name of the queue. One queue name is set once for each pool. Each queue that is created, after the first one, is comma delimited.
- **MaxQueues** – Maximum number of queues in a pool. The MaxQueues value is set once for each pool.
- **Overflow** – Number of messages to be processed by the current queue before the next queue is used. The overflow value is set once for each pool.
- **DefaultPool** – Default pool for data transfers. This default value is used if no pool is specified with the request.

For a pool to work correctly, these settings must be included in the FTF configuration files for both the FTF Sender and the FTF Receiver involved in the data transfer.

This section describes how to set the FTF configuration file values for the case study.

### Setting the Pool Name

The Data Pools section of the FTF configuration file is used to specify the names for all pools created for the current data-transfer request. The case study requires only one pool to be set up. For the purposes of demonstration, a second pool will be added to the FTF configuration file to handle processing other than that described in this chapter.

## Using Pools

### Implementing the MQSeries Objects

The following pool name could be used for the case study:

```
POOLS :  
name=POOL1, POOL2
```

In this example, POOL1 will be configured to support the case study. POOL2 is used for other processing.

### Setting the Pool Detail Information

After you have established the pool names, you must establish the detail information for each pool. Detail information determines the name of the queues used in the pool, the number of queues, and the overflow value. In this example, detail information is established for POOL1.

```
POOL1 :  
QueueName=poolQueue  
MaxQueues=5  
Overflow=2000
```

In this example, FTF uses the FTF Receiver local queues already set up. The dynamic queue name expansion facility defines the pool as including the pools *poolQueue*, *poolQueue.1*, *poolQueue.2*, *poolQueue.3* and *poolQueue.4*, the local queues that have already been established.

### Using Alternative Naming Conventions

In some cases, you may want to use a naming convention other than the one supported by the dynamic queue name expansion facility. To use an alternate naming convention, specify each queue's name in the QueueName stanza. In the following example, the QueueName stanza is used to create an alternate naming convention for the case study:

```
QueueName=pool1, pool1a, pool1b, pool1c, pool1d
```

In order for pools to work with these queue names, the local queues set up on the FTF Receiver must have been set up with the same names.

The MaxQueues stanza determines that the pool contains five queues. You should set this stanza to a value equal to the number of local queues established on the FTF Receiver. (If you do not want to use all the local queues set up on the FTF Receiver, you can also set this stanza to a lesser value.) The number of channels does not have to equal the number of MaxQueues.

The Overflow value determines how many messages are received by a pool queue before the next queue is used. You should set this value based on the message size component set in the data-transfer request and your connections' throughput capacity. If a queue becomes full because its MAXDEPTH value has been exceeded or because queue space has been used up, FTF moves to the next queue in the pool.

In this case, FTF uses the default message size of .5 MB, meaning that each file consists of 2000 messages. Because of the throughput capacity of the T1 lines, you can start by setting the overflow to 2000. To attain the most efficient performance for FTF, you may have to adjust the message size and overflow values to account for installation-specific factors, such as network reliability and other traffic on the connections.

### **Overflow and Load Balancing**

You can also use the overflow stanza to balance the data-transfer load over each channel. In the case study, FTF is sending a 2GB file from the FTF Sender across five channels to the FTF Receiver. To balance the load effectively across all five channels requires FTF to send 400 MB across each channel.

If the FTF message size is left at its default value of .5 MB, sending 400 messages across each channel would result in load balancing. To achieve this configuration, set the overflow value to 400.

### **Setting the Default Pool**

The default pool setting determines which pool is used by FTF if no pool is specified in the interface. In the case study, this data transfer being described is performed more often than any other data transfer. As a result, the default pool is set to POOL1.

## Example

The following figure contains the entire Data Pools section of the FTF configuration file that was created for the case study's data transfer.

```
POOLS:
name=POOL1, POOL2

POOL1:
QueueName=poolQueue
MaxQueues=5
Overflow=2000

POOL2:
QueueName=QM2L1, QM2L2, QM2L3, QM2L4
MaxQueues=4
Overflow=2

DEFAULT POOL:
name=POOL1
```

## Specifying a Pool

While the FTF configuration file includes a default pool, used for processing if no pool is specified with the data transfer request, each FTF interface allows you to override this default by specifying a pool. This section lists instructions for overriding the default pool specification in each FTF interface.

In each case, the pool specified in the interface must have been defined in the FTF configuration file. If the pool is not defined in the FTF configuration file, the FTFRCE\_INVALID\_POOL\_DEFINITION error is generated and the data-transfer request fails.

## Command-Line Interface

The FTF command executes a data-transfer request from the command-line interface. To override the default pool setting in the FTF command, use the `-pool` argument. In the following example, `PRODPOOL2` is specified as the pool to use during the data-transfer request.

```
FTF -lqm PROD16A -oqm PROD16A -sqm PROD16A
-dqm PROD7B -pool PRODPOOL2
```

For complete information about using the FTF command, see “FTF” in the *Tivoli Data Exchange Technical Reference*.

## C API

The `FTFReq` function executes a data-transfer request from the FTF C API. This function has the following prototype:

```
#include "ftfc.h"

FTFVOID FTFReq (MQHCONN      hQM,
                FTFRequestMsgInfo *pRequestInfo,
                FTFCA          *pftfca);
```

The `FTFRequestMsgInfo` data type is a data structure that contains the `FTFJobInfo` data structure. The `FTFJobInfo` data structure contains a data element called *poolName*. To override the default pool, specify a pool name in this data element.

For more information about executing a data-transfer request from the C API, see the following topics:

- “FTFReq” in the *Tivoli Data Exchange Technical Reference*
- “FTFJobInfo” in the *Tivoli Data Exchange Technical Reference*
- “FTFRequestMsgInfo” in the *Tivoli Data Exchange Technical Reference*

## **COBOL API**

In the FTF-JOB-INFO field segment of the FTF-REQUEST-MESSAGE-INFO COBOL data structure, a data element named FTFJ-POOL-NAME allows you to specify the pool name. For more detailed information about the FTF-REQUEST-MESSAGE-INFO data structure, see “FTF-JOB-INFO” and “FTF-REQUEST-MESSAGE-INFO” in the *Tivoli Data Exchange Technical Reference*.

## **ISPF Interface**

The ISPF interface allows you to perform all FTF interface functions from interactive text panels on OS/390 systems. For a complete description of how to use the ISPF panels to execute a data-transfer request, see “Requesting a Data Transfer” on page 129 found in the “Using the ISPF User Interface” chapter. Setting the pool name is discussed on page 341.

## **5250 Interface**

The 5250 interface allows you to perform all FTF interface functions from interactive text panels on AS/400 systems. For a complete description of how to use the 5250 panels to execute a data-transfer request, see the Data Pool Name attribute that is discussed in the 5250 interface chapter (page 167).

## **FTF GUI Interface**

The FTF GUI allows you to perform all FTF interface functions from a Java-based GUI. For a complete description of how to use the FTF GUI to execute a data-transfer request, see the Data Pool Name attribute discussed in “Using the FTF GUI” chapter.



---

# Setting Up Configuration Queues

Tivoli Data Exchange (TDE) includes the capacity to allow each node on the system to operate with configuration data stored in a queue rather than with a file. This chapter describes how to set up and work with a configuration queue.

It contains the following sections:

Section	Page
Setting the FTF Configuration Queue Stanza	348
Loading the Configuration Queue	349
Using the Configuration Queue	350

## Assumptions

This chapter makes the following assumptions:

- You understand the roles of the TDE (FTF) Manager, FTF Receiver, and FTF Sender.
- You understand how queues are used in FTF and MQSeries.

## Overview to Setting Up a Configuration Queue

Setting up a queue to hold configuration information lets you use an MQSeries queue to hold processing variables rather than keeping the data in a file. To use a configuration queue in FTF operations, you must perform the following steps:

1. Specify the configuration queue. (See “Setting the FTF Configuration Queue Stanza” on page 348.)
2. Load a configuration queue. (See “Loading the Configuration Queue” on page 349.)
3. Use a configuration queue in a FTF operation. (See “Using the Configuration Queue” on page 350.)

## Setting the FTF Configuration Queue Stanza

Generally, the FTF configuration file holds the values by which the FTF system operates. The `FTFConfigQueue` stanza defines the name of an MQSeries queue where the configuration file information is to be stored. This stanza is inserted in the FTF configuration file with the following properties:

**FTFConfigQueue** = *ConfigQueueName*

Where *ConfigQueueName* is the name of the MQSeries queue that will hold configuration information. You must create this queue in MQSeries before you can use the `FTFCFG` command to populate it. When the `FTFCFG` command is used to populate the configuration queue, the queue name in the `FTFConfigQueue` stanza receives the configuration file contents. For more information about the FTF configuration file, see “FTF Configuration” in the *Tivoli Data Exchange Installation Guide*.

### Example

In the following example, the value in this stanza — `FTFCONFQ` — is an MQSeries queue name which is defined on each of the nodes that are to be used by FTF.

```
FTFConfigQueue=FTFCONFQ
```

## Loading the Configuration Queue

The FTFCFG command loads the configuration file onto the configuration queue. The format of the FTFCFG command follows:

```
FTFCFG -lqm localQueueMgr -node nodeName -nodefile nodeFilename -cfile  
configFile -ofile optionsFile -version [-help| -h | -?]
```

Where:

- **-lqm *localQueueMgr*** – Determines the queue manager to which the FTFCFG command is attached. This value is required on OS/390 systems. On all other systems, if it is not specified, the FTFCFG command connects to the default queue manager specified in the MQSeries configuration.
- **-node *nodeName*** – Lists the FTF nodes that require updating. This argument can be specified more than once, and can be used with -nodefile. Duplicates are eliminated. On the AS/400 platform, this argument may be used only once. To specify multiple nodes, use the -nodefile argument.
- **-nodefile *nodeFilename*** – Specifies a file which lists the FTF nodes that require updating. This argument may be used with the -node argument. Duplicates are eliminated.
- **-cfile *configFile*** – Can contain the fully qualified path and file name for the FTF configuration file. This value must be set on OS/390. In operating systems other than OS/390, setting the FTF\_CONFIG\_FILE environment variable is optional.
- **-ofile *optionsFile*** – Contains the fully qualified path and file name of a text file used to contain command-line arguments for the FTFCFG command. In the options file, you can set any of the command-line arguments that can be set for the FTFCFG command. Any values specified on the command line override the values in the options file.
- **-version** – Returns the current FTF version, release, and patch number. Any command entered with this argument ignores all other arguments and returns the version information.
- **-help, -h, or -?** – Displays a list and description of the FTFCFG command's command-line arguments.

## Setting Up Configuration Queues

### *Using the Configuration Queue*

The FTFCFG command specifies various nodes either through the `-node` or `-nodefile` arguments. The command causes the configuration file information to be loaded in the configuration queue specified in the FTF configuration file's FTFCfgQueue stanza. In the example above the queue is FTFCNFQ.

The following example of the FTFCFG command shows a configuration queue being loaded. Its configuration file is `\opt\ftf\ftfconfig.ini`. The node being loaded is PROD11A.

```
FTFCFG -node PROD11A -cfile \opt\ftf\ftfconfig.ini
```

When using FTF with an ISPF or a 5250 interface, enter the FTFCFG command at the command line.

## Using the Configuration Queue

Using a configuration queue allows you to store configuration information in a queue rather than in a file. Each node in an enterprise can have its own configuration queue. The configuration file can be edited at a central location by the FTF administrator and loaded to each node with the FTFCFG command.

After the configuration queue is loaded, FTF can operate using either a configuration queue or a configuration file. With the exception of the FTFCFG command, each FTF command that accepts the `-cfile` argument can specify the `-cq` argument. The `-cq` argument has the following format:

- **-cq *configQueueName*** - Designates the queue from which the configuration information is to be retrieved for this FTF instance on this node. The `-cq` argument points FTF to the queue name rather than to the standard configuration file.

If both the `-cfile` and the `-cq` arguments are specified in the same command, the `-cq` argument takes precedence. If the `-cq` argument is present and specifies the queue name, then FTF scans the queue for its configuration information.

The FTF command displayed below specifies a configuration queue of FTFCNFQ to provide the values required to transfer the file `ftf\ftfdata.txt`. The FTF command requires other arguments to run correctly, the example is coded as it is for simplicity and example. For a full description of the FTF command, see "FTF" in the *Tivoli Data Exchange Technical Reference*.

```
FTF -cq FTFCNFQ -spath ftf\ftfdata.txt
```

## Setting Environment Variables

Setting the FTF\_CONFIG\_QUEUE environment variable removes the requirements for specifying the -cq value with each FTF command. Each platform allows a variable to be set before executing a command or set of commands. Usually the FTF Administrator sets this variable for you. In other cases you can set this variable yourself. As an example of setting the environment variable in the Windows NT platform you would key at the command line:

```
SET FTF_CONFIG_QUEUE=FTFCONFQ
```

In this example, the FTFCONFQ configuration queue is used for any FTF command for which the -cq and -cfile arguments are not specified. This feature allows you to have a default queue, but override it with another queue. This would be especially useful in testing or debugging.

## Changing Configuration Queue Information

Each time you change the values in the configuration queue, you must perform the following set of steps:

1. Edit and change the values in the configuration file.
2. Use the FTFCFG command to load the updated configuration queue.
3. Use the appropriate FTF commands with the -cq argument to reference the changed values in the queue.

## Multiple Instances of FTF

The steps described to this point of the chapter are for getting one FTF instance to run using a configuration queue. Each instance has only one configuration queue or configuration file from which it operates.

You can run multiple FTF instances. Often, a new instance is used to test new releases while maintaining another instance of a production environment. There are other situations where you might establish more than one instance of FTF. You can set up a different instance, each having a different configuration queue or file.



---

---

# Logging Tivoli Data Exchange Information

Each Tivoli Data Exchange (TDE) component writes to a local log file. TDE also contains a log queue that can receive log messages from specified TDE components. This chapter describes how to use both logging options to manage log information written by TDE components during file transfers.

It contains the following sections:

Section	Page
Logging Tivoli Data Exchange Information	353
Writing to a Log File	354
Using the FTFLOG Process	356

## Assumptions

This chapter makes the following assumptions:

- You understand the roles of the TDE Manager, TDE Receiver, and the TDE Sender.
- You understand how queues are used in TDE.

## Tivoli Data Exchange Logging Options

TDE allows you to log information using two different methods, a local log file you specify when you start a TDE component and/or an FTFLOG daemon process.

In both cases, the log contains the following information:

- Messages generated when TDE components start up
- Error message
- Status messages

## Writing to a Log File

When you start a TDE component, you can specify a local log file where the component will write log information. The *-lfile* argument for the FTFMGR (TDE Manager), FTFRCV (TDE Receiver), and FTFSDR (TDE Sender) commands specifies the log file.

---

### Note:

- OS/390 does not support writing to log files. On OS/390, log information is written to the standard SYSOUT queue.
- 

The following example shows a TDE Manager being started. Its log file is `\opt\ftflog\mgr.log`.

```
FTFMGR -lqm PROD11A -cfile \opt\ftf\ftfconfig.ini  
-lfile \opt\ftflog\mgr.log
```



The same syntax applies to the FTFRCV and FTFSDR commands.

### **Local Log Files and the TDE Service**

To specify a local log file for a TDE component started by the TDE Win 32 service component, you must modify the FTF.INI file on the machine on which the service is being started.

Each FTF.INI file contains an FTFStart section that determines the number of TDE instances being started and environment sections that specify command-line arguments for each component started as part of the instance. To create a log file for a component, you must specify the log file in the component's command line.

For complete information about setting log file values in the Win 32 service, see "Configuring Clients" in the *Tivoli Data Exchange Installation Guide*.

## **Removing Log Information from Standard Output**

On UNIX platforms, you can start TDE components as background processes. The command window from which you issue the component startup commands allows you to perform other activities, but it still receives log information in the form of standard output.

To eliminate log information from standard output, redirect the command's standard output to /dev/null.

In the following example, the TDE Manager is started as a background process. A log file is specified and its standard output is redirected to /dev/null.

```
FTFMGR -lqm PROD11A -cfile /opt/ftfmq/config.ini  
-lfile /opt/ftfmq/ftfmgr.log >/dev/null&
```

## Log File Contents

The following figure contains a sample of a log file's content. In this case, the log file contains several entries listing the queues on which processing took place, as well as status messages about the TDE Manager component and completion messages about specific data transfers.

```
1999/01/29 09:09:07 Operating on control/input queue FTFMGR.CONTROL
1999/01/29 09:09:07 Operating on synchronization queue FTFMGR.SYNC
1999/01/29 09:09:07 Connecting to local queue manager TESTHP
1999/01/29 09:09:07 TDE Manager initialization complete
1999/01/29 09:09:43 Processing shutdown command from ftfusr1 at TESTHP
1999/01/29 09:09:43 TDE Manager termination complete
1999/01/29 09:09:51 Operating on control/input queue FTFMGR.CONTROL
1999/01/29 09:09:51 Operating on synchronization queue FTFMGR.SYNC
1999/01/29 09:09:51 Connecting to local queue manager TESTHP
1999/01/29 09:09:51 TDE Manager initialization complete
1999/01/29 09:46:21 FTF request complete 5f091901-b789-11d2-9774-ce3ca2fd09dc
1999/01/29 09:51:08 FTF request complete 0b4aff81-b78a-11d2-a293-81e4459355eb
1999/01/29 09:57:20 FTF request complete e905b181-b78a-11d2-919a-dad3a76b0465
1999/01/29 09:57:30 FTF request complete eefb9281-b78a-11d2-89df-c7e005339c0f
1999/01/29 09:58:56 FTF request complete 223e2181-b78b-11d2-8b80-99234e6a4a47
1999/01/29 09:59:00 FTF request complete 24a07b81-b78b-11d2-9fef-866269daaf51
```

## Using the FTFLOG Process

Using the FTFLOG process allows for centralized logging of the TDE components. It can also allow for easier remote access to log information.

You must perform the following tasks to use the FTFLOG process to receive log information:

- Configure a log queue in the TDE configuration file (see page 357)
- Start the FTFLOG component

## Configuring a Log Queue

Writing to a log queue directs log information to a queue specified in the TDE configuration file. When a TDE component is started using that TDE configuration file, that component writes to the specified log queue.

To write to a log queue, set the following stanza in the TDE configuration file *Log Queue* properties:

```
FTFLogQueue = logQueue1[, logQueue2,...logQueueN]
```

Where *logQueue1*, *logQueue2*, and *logQueueN* are the queues to which log messages are written. If you want a queue to accept logging information, you must specify at least one log queue name.

### Example

In the following example, the FTFLogQueue stanza is set to write log messages to the following queues: PROD1.LOG, PROD6.LOG, and PROD9.LOG.

```
FTFLogQueue=PROD1.LOG,PROD6.LOG,PROD9.LOG
```

## Retrieving Information from Log Queues

To retrieve messages from a log queue, you must start the FTFLOG component. FTFLOG is a daemon process that gets the log messages currently residing in the specified log queue and writes the messages to log files.

The FTFLOG command has the following syntax:

```
FTFLOG -lqm localQMGr -ldir targetDir -q logQueue
```

Where:

- *localQMGr* is the MQSeries queue manager that the FTFLOG process connects to. This value is required on OS/390 systems. On all other systems, if this value is not specified, the command connects to the default queue manager set in the MQSeries configuration.
- *targetDir* is the name of the directory to which the log files are written.
- *logQueue* is the name of the log queue from which the log information is taken.

## Logging Tivoli Data Exchange Information

### Using the FTFLOG Process

One log file is created for each component that writes to the specified log queue. The naming convention for the log files is:

*queueName.componentName.log*

Where:

- *queueName* is the name of the log queue
- *componentName* is SDR for a TDE Sender, RCV for a TDE Receiver, and MGR for a TDE Manager.

If the specified log files already exist, the new log information is appended to them.

## Setting OS/390 Options

On OS/390 platforms, the FTFLOG command must allocate space for the log files it creates.

### Specifying an Esoteric Unit Name

To specify an esoteric name for the OS/390 UNIT value, follow these steps:

- Do not set a value in the MVSVOLUME stanza in the TDE configuration file.
- Specify the *Unit* value in the **-unit** argument or in the MVSUNITNAME stanza in the TDE configuration file on either the TDE Sender or the TDE Receiver.

In order to allocate space on OS/390, FTFLOG supports the following arguments:

FTFLOG -org *fileOrg* -recfmt *recordFormat* -lrecl *logRecLength* -blksize *blockSize* -unit *unitName* -volser *serialNumber* -alcunit *allocUnit* -primary *primAlloc* -secondary *secAlloc* -dirblks *numBlks*

Where:

- **-org** *fileOrg* – Determines the file organization of the log files on OS/390. **Valid values:** Physical Sequential (PS), Partitioned Data Set (PDS)

- **-recfmt** *recordFormat* – Determines the record format for the log files on OS/390. **Valid values:** F (fixed), V (variable), FB (fixed block), and VB (variable block).
- **-lrecl** *logRecLength* – Determines the logical record length for the log files on OS/390. **Valid values:** 1-32767
- **-blksize** *blockSize* – Determines the block size for the log files on OS/390. **Valid values:** 0-32767
- **-unit** *unitName* – Determines the unit name for the log files on OS/390. This argument's value is installation-dependent. Obtain it from your OS/390 administrator.
- **-volser** *serialNumber* – Determines the volume serial number for the target on OS/390. This argument's value is installation-dependent. Obtain it from your OS/390 administrator.
- **-alcunit** *allocUnit* – Determines the allocation unit used for the target on OS/390. **Valid values:** CYL (cylinder), BLK (block), and TRK (track).
- **-primary** *primAlloc* – Determines the number of primary allocation units required on OS/390.
- **-secondary** *secAlloc* – Determines the number of secondary allocation units required on OS/390.
- **-dirblks** *numBlks* – Sets up the number of directory blocks used in allocating the target PDS if it does not exist. If this argument is not specified, the value in the TDE configuration file is used. If neither is specified, the PDS allocation fails. **Valid values:** 1-32767 **Default value:** 10

### **Example 1 – Open Systems Environment**

In the following example, a TDE Manager, Receiver, and Sender are running on the PROD1 and PROD2 queue managers. The configuration file governs both sets of components and includes the following stanza:

```
FTFLogQueue=PROD1_2.QLOG
```

The PROD1\_2.QLOG is defined as a local queue on the PROD1 queue manager and is defined as a remote queue on the PROD2 queue manager.

**Logging Tivoli Data Exchange Information**  
*Using the FTFLOG Process*

The following command is run to start the FTFLOG daemon to log information from both nodes:

```
FTFLOG -lqm PROD1 -ldir c:\ftf\log -q PROD1
```

When the FTFLOG daemon is started, it creates the following log files and writes log information to them:

- PROD1 Manager – PROD1.MGR.LOG
- PROD1 Sender – PROD1.SDR.LOG
- PROD1 Receiver – PROD1.RCV.LOG
- PROD2 Manager – PROD2.MGR.LOG
- PROD2 Sender – PROD2.SDR.LOG
- PROD2 Receiver – PROD2.RCV.LOG

**Example 2 – OS/390 Environment**

In the second example, the values listed in the following table must be set in the FTFLOG command.

Value	Setting
File Organization	Physical Sequential
Record Format	Fixed Block
Logical Record Length	255
Block Size	65,535
Unit Name	SYSALLDA
Volume Serial Number	TECH01
Allocation Unit	TRK
Primary Allocation	15
Secondary Allocation	30

The following FTFLOG command requests a file transfer with the specified settings. Required command-line options that are not listed are specified in the options file. The lqm is PROD1, the log directory is FTF.LOG, and the log queue is PROD1.

```
FTFLOG -ofile FTFMQ.FTF.OPT -org PS -recfmt FB  
-lrecl 255 -blksize 65535 -unit EDISource  
-volser 3144312 -alcunit BLK -primary 15  
-secondary 30 -lqm PROD1 -ldir FTF.LOG -q PROD1
```

## Stopping the FTFLOG Daemon

To stop the FTFLOG daemon, enter the following command at a command prompt:

```
FTFEND -lqm localQMgr -cpt 4
```

Where *localQMgr* is the local queue manager that the FTFLOG daemon is connected to.

For a complete description of the FTFEND command, see “FTFEND” in the *Tivoli Data Exchange Technical Reference*..

---

### Note:

When you submit an FTFEND request to shut down TDE component(s) on an lqm, a shutdown message is placed on the queue defined as the FTFLogQueue in the TDE configuration file. If this queue is defined as a remote queue, the FTFLOG process running on the remote queue manager is shut down.

---

## Example

In the following example, the FTFLOG daemon connected to PROD1 is shut down.

```
FTFEND -lqm PROD1 -cpt 4
```





# Index

## Numerics

- 4690
  - file distribution types 317– 318
  - FTF command 330
  - FTFEND command 331
  - FTFMGR command 330
  - FTFPING command 330
  - FTFRCV command 330
  - logical filenames 317
  - user exit 329
- 4690 installation
  - initialization 318
- 5250 interface
  - description 167
- 5250 panels
  - FTF 173
  - FTF Cancel 196
  - FTF End 204
  - FTF Ping 195
  - FTF Start Manager 212
  - FTF Start Receiver 217
  - FTF Start Sender 214
  - FTF Status 199
  - logon 168
  - main menu 169

## A

- accessing ISPF panels 129
- alcunit argument
  - FTFLOG 359

- all argument
  - in AS/400 208
- allocation options
  - in ISPF file transfer 145
- allocation unit 359
  - in GUI file transfer 245
  - in ISPF file transfer 147
- append mode 282
- appending data to an existing file 282
- architectural overview 21
- argument options
  - in ISPF file transfer 133– 134, 136– 137
- arguments 71, 73– 74, 76, 78, 80, 82, 91
  - FTFCFG 349– 350
  - FTFDIRMN 311– 312
  - FTFEND 47– 48
  - FTFLOG 357– 359
  - FTFMGR 39
  - FTFRCV 41
  - FTFSDR 40
  - FTFSTART 33, 39– 41, 46
- AS/400
  - arguments 173– 182, 184– 193, 195– 205, 207– 224
  - FTF 173– 182, 184– 193
  - FTF Cancel 197– 198
  - FTF Cancel panel 199
  - FTF Config 210– 211
  - FTF End 204– 205
  - FTF Logger 222– 224
  - FTF Manager 212– 213
  - FTF panel 178
  - FTF Receiver 218– 221
  - FTF Sender 214– 217

## B

- FTF Stage 207–209
- FTF Status 200–203
- FTF Status panel 200
- FTFPing 195–196
- logon 168
- priority 196
- STRFTFMGR 213
- using the menu 36
- AS/400 exits
  - protected job submission 91
- AS/400 file type 185
  - in ISPF file transfer 149
- AS/400 logical file
  - data conversion 106
  - specifying as destination 109
  - specifying as source 109
  - transferring 106
  - write only to first record format 106
- AS/400 logical file data
  - receiving from a fixed-length text file 115
  - sending 109
  - sending to AS/400 logical files 119
  - sending to delimited text file 113
  - sending to fixed-length text file 112
- AS/400 logical file data transfer
  - getting status 121
  - modifying configuration file 107
  - specifying the file type 107
- AS/400 physical file
  - creating destination file 117, 119
  - data conversion 106
  - specifying as destination 109
  - specifying as source 109
  - transferring 106
- AS/400 physical file data
  - receiving from a fixed-length text file 115
  - sending 109
  - sending to AS/400 logical files 117
  - sending to AS/400 physical files 117
  - sending to delimited text file 113
  - sending to fixed-length text file 112
- AS/400 physical file data transfer
  - getting status 121
  - modifying configuration file 107
  - receiver status error messages 123
  - sender status error messages 122
  - specifying the file type 107
- AS/400 record format
  - specifying in a transfer 110
- AS/400 service programs
  - compiling and generating 90
- AS/400 table name
  - specifying in file transfer 110
- authorization entry 216, 220
- authorization exit 216, 220
  - associated data structures 102
  - command line 102
  - description 93, 102
  - installation 96
  - reference 102
  - sample exits 103
  - usage 94
  - usage notes 103
- authorization exit enabled 216, 220
- authorization module
  - customizing 94
  - data structures 94
  - description 94
  - OS/390 version included 94
  - return codes 95
- authorization module function
  - description 100
  - parameters 100
  - related data structures 101
  - return value 100
  - sample shell 101

## B

- blksize argument
  - FTFLOG 359
- block count
  - in GUI file transfer 244
- block size 359
  - in GUI file transfer 245
  - in ISPF file transfer 147

bufno argument  
FTF 147

## C

cancel an in-flight request panel 160  
cancel mode 177

in ISPF file transfer 136

cancellation

enabling in GUI 247

cancelling a data-transfer request

in ISPF panels 160

Cannot create the target table without source table  
schema error

AS/400 logical file transfer 124

AS/400 physical file transfer 124

CCSID 185

in ISPF file transfer 149

cfile argument

FTFCFG 349

FTFDIRMN 311

FTFEND 47

FTFMGR 39

FTFRCV 41

FTFSDR 40

FTFSTART 33

channels

designing 338

command area

ISPF panels 128

complete all work 205

component 47

component to end 205

component type

in GUI component shut down 258

in ISPF component shut down 164

Components

types 28

components 21

shutting down 46

starting individually 38

starting on AS/400 36

starting on OS/390 33

starting on Windows NT 34

components frame 234

config file 207, 210, 222

config file argument 195, 200, 204, 212, 214,  
218

in AS/400 197

config queue 207, 212, 215, 222

config queue argument 195, 200, 205, 218

in AS/400 198

configuration file 175

FileTypes property 106–107

FTFCFG 349

FTFEND 47

FTFLogQueue stanza 357

FTFMGR 39

FTFRCV 41

FTFSDR 40

FTFSTART 33

in GUI file transfer 239

modifying for AS/400 logical file transfer  
107

modifying for physical file transfer 107

configuration queue

in FTF Request 176

configuration queue name

FTFCFG 350

FTFEND 47

FTFMGR 39

FTFRCV 41

FTFSDR 40

FTFSTART 33

configuring a log queue 357

connector data 217, 221

connector entry 217, 221

connector exit 217, 220

connector exit enabled 220

Contacting Customer Support 17

conventions 18

cpt argument

FTFEND 47

cq argument

FTFCFG 350

FTFDIRMN 311

## E

- FTFEND 47
- FTFMGR 39
- FTFRCV 41
- FTFSDR 40
- FTFSTART 33
- create library 185
- create mode 282
- creating a destination file 282
- current Status Message frame 234

## D

- data compression
  - in GUI data transfer 242
  - in ISPF file transfer 137
- data persistence
  - in GUI file transfer 242
  - in ISPF file transfer 137
- data pool name 180
- data transfer
  - appending data to an existing file 282
  - creating a destination file 282
- data-transfer request
  - cancelling 160
- date
  - in ISPF status information 158
- date range
  - in GUI status request 231
  - in ISPF status request 156
- Ddata
  - in ISPF file-transfer request 152
- ddata parameter
  - syntax 109
- DefaultPool stanza 341
- Defining the Command Line for Status Offload 63
- Delete the Source 182
- delete the source file 182
- deleting status information
  - GUI 236
  - ISPF panels 158
- delimited file

- specifying in AS/400 transfer 110
- delimiter character
  - specifying in AS/400 transfer 110
- destination data
  - specifying 109
- destination file
  - creating 282
- destination file data 175
- destination file type 175
  - specifying 109
- destination path 174, 203
  - in ISPF file transfer 130
- destination queue manager 174, 195, 202
- dirblks argument 359
- directory blocks 359
  - in ISPF file transfer 146
- display FTF version 178, 196
  - AS/400 199
- display version 203, 205, 209, 211–212, 215, 219, 223
- displaying log information
  - ISPF panels 158
- displaying summary information
  - ISPF panels 158
- DLM 110
  - specifying a source portal 152, 174–175, 242, 244
- dqm
  - and stageonly 181
  - in GUI file transfer 239
  - in GUI ping 252
  - in GUI status request 232
  - in ISPF file transfer 130
  - in ISPF ping 162
  - in ISPF status request 156
- Dtype
  - in ISPF file-transfer request 152
- dynamic queue name expansion facility 339

## E

- earliest date of transfer 200

- enable compression 179
- enabling cancellation
  - in GUI file transfer 247
- eNDI Client for 4690
  - 4690 initialization 318
- eNDI Client for 4690 Receiver
  - starting 320
- Entry Point
  - in ISPF file-transfer request 151
- entry point of DLL 71, 74, 76, 78, 80, 82
- environment variables
  - setting 351
- esoteric unit name, setting 146, 180, 245, 358
- event
  - ISPF status information 158
- exit 71, 73, 76, 78, 80, 82
  - associated data structures 102
  - Manager post-process 73
  - Manager pre-process 71
  - Receiver post-process 80, 82
  - Sender post-process 78
  - Sender pre-process 76
- exit argument 73, 76, 78, 80, 82
  - exit number 71, 73, 76, 78, 80, 82
- exit data
  - in GUI file transfer 250
  - in ISPF exit data 141
- exit DLL
  - in GUI file transfer 250
  - in ISPF file transfer 141
- exit entry
  - in GUI file transfer 250
  - in ISPF file transfer 141
- exit number
  - in GUI file transfer 249
  - in ISPF file transfer 140
- exitdata 71, 74, 76, 78, 80, 82
- exitdata argument 71, 74, 76, 78, 80, 82
- exitdll 71, 73, 76, 78, 80, 82
- exitdll argument 71, 73, 76, 78, 80, 82
  - DLL name 73, 76, 78, 80, 82
- exitentry 71, 74, 76, 78, 80, 82
- exitentry argument 71, 74, 76, 78, 80, 82
- exits

- AS/400 91
- expiry
  - description 30
  - in GUI file transfer 246
  - in ISPF file transfer 136
- expiry time 184

## F

- fail argument option
  - in ISPF file transfer 133
- Failed to create target table based on the source
  - table schema error
    - AS/400 logical file transfer 125
    - AS/400 physical file transfer 125
- Failed to load data into target table
  - AS/400 logical file transfer 125
  - AS/400 physical file transfer 125
- Failed to receive source table schema
  - AS/400 logical file transfer 124
  - AS/400 physical file transfer 124
- Failed to retrieve DTYPE information error
  - AS/400 logical file transfer 123
  - AS/400 physical file transfer 123
- Failed to retrieve STYPE information
  - AS/400 logical file transfer 124
  - AS/400 physical file transfer 124
- Failed to retrieve table column information
  - AS/400 logical file transfer 123
  - AS/400 physical file transfer 123
- Failed to retrieve table data error
  - AS/400 logical file transfer 123
  - AS/400 physical file transfer 123
- Failed to retrieve target table schema
  - AS/400 logical file transfer 125
  - AS/400 physical file transfer 125
- Failed to send table schema error
  - AS/400 logical file transfer 123
  - AS/400 physical file transfer 123
- file ASP
  - in ISPF file transfer 149
- file Aux Storage Pool

## F

- in AS/400 185
- file containing options 203
- file distribution types
  - 4690 317– 318
- file organization
  - in GUI file transfer 244
  - in ISPF file transfer 146
- file text 185
  - in ISPF file transfer 149
- file type 178
  - in GUI file transfer 242– 243
  - in ISPF file transfer 134
- file-transfer request
  - AS/400 173
  - executing in ISPF panels 131
  - expiry 30
  - from ISPF panels 129
  - resubmitting in ISPF panels 158
- FileTypes property 106– 107
  - example 108
- filter criteria
  - loading 230
- filtering status information
  - in the GUI 229
- fixed-length file
  - specifying in AS/400 transfer 110
- FIXLEN
  - specifying a source portal 152, 174– 175, 242, 244
- fnotify type
  - in ISPF 145
- format 200
- from stage 181
- FTF
  - AS/400 173
  - bufno 147
  - command 129
- FTF Cancel panel
  - config file 197
  - config queue 198
  - display FTF version 199
  - FTF transaction identifier 197
  - local queue manager 197
  - options file 198
  - originating queue manager 198
- FTF command
  - append mode 282
  - create mode 282
  - ddata parameter 109
  - dtype parameter 109
  - noreplace mode 282
  - sdata parameter 109
  - selection 170
  - stype parameter 109
  - using for AS/400 logical file transfers 108
  - using for AS/400 physical file transfers 108
- FTF components host node 205
- FTF Config panel
  - config file 210
  - display version 211
  - local queue manager 210
  - node 210
  - node file 210
  - options file 210
- FTF End panel
  - complete all work 205
  - component to end 205
  - config file 204
  - config queue 205
  - display version 205
  - FTF components host node 205
  - local queue manager 204
  - seconds to wait for response 205
  - terminate with pending work 205
- FTF ID
  - in GUI status request 231
  - in ISPF cancel request 161
  - in ISPF status request 155
- ftf id argument
  - in ISPF staged request 154
- FTF Logger panel
  - config file 222
  - config queue 222
  - display version 223
  - job description 223
  - job name 223
  - job queue 224
  - local queue manager 222

- log directory 222
- log file 223
- logger queue 222
- options file 223
- FTF Manager panel
  - config file 212
  - config queue 212
  - display version 212
  - job description 213
  - job name 213
  - local queue manager 212
  - log file 213
  - node name 213
- FTF on 4690 330
- FTF panel
  - AS/400 file type 185
  - CCSID 185
  - configuration file 175
  - configuration queue 176
  - create library 185
  - data pool name 180
  - destination file data 175
  - destination file path 174
  - destination file type 175
  - destination queue manager name 174
  - enable compression 179
  - expiry time 184
  - file aux storage pool 185
  - file text 185
  - file type 178
  - from stage 181
  - FTF cancel mode 177
  - FTF transaction label 177
  - FTF transfer id 182
  - library aux storage pool 185
  - library text 185
  - local queue manager 175
  - make dest directory 178
  - mgr post data 189
  - mgr post exit 188
  - mgr pre data 188
  - mgr pre entry 188
  - mgr pre exit 187
  - mode of writing file 178
  - MVS ALCUNIT 187
  - MVS BLKSIZE 187
  - MVS GDG model 187
  - MVS LRECL 187
  - MVS org 186
  - MVS primary allocation 187
  - MVS RECFMT 187
  - MVS secondary allocation 187
  - MVS UNIT 187
  - MVS VOLSER 187
  - options file 176
  - originating queue manager 177
  - override max msq size 181
  - pad records 177
  - padding character 177
  - receiver portal data 193
  - receiver portal entry 193
  - receiver portal exit 193
  - receiver post data 192
  - receiver post entry 191
  - receiver post exit 191
  - receiver pre data 191
  - receiver pre entry 191
  - receiver pre exit 190
  - record length 185
  - record wrapping 178
  - seconds to wait for reply 178
  - sender portal data 192
  - sender portal entry 192
  - sender portal exit 192
  - sender post data 190
  - sender post entry 190
  - sender post exit 190
  - sender pre data 189
  - sender pre entry 189
  - sender pre exit 189
  - source file data 173
  - source path 173
  - source queue manager name 173
  - source type 173
  - stage only 181
  - stage persistant 179
  - stage the source 181
  - transfer priority 181

## F

- transmission persistent 179
- trusted transfer 179
- FTF Ping Dialog window 251
- FTF Receiver and the Status Subsystem 45
- FTF Receiver panel
  - authorization entry 220
  - authorization exit 220
  - authorization exit enabled 220
  - config file 218
  - config queue 218
  - connector data 221
  - connector entry 221
  - connector exit 220
  - connector exit enabled 220
  - display version 219
  - job description 221
  - job name 221
  - job queue 221
  - local queue manager 218
  - log file 219
  - node name 219
  - options file 219
  - sync queue override 218
- FTF Receiver Processing Summary 32
- FTF Sender panel 215
  - authorization entry 216
  - authorization exit 216
  - authorization exit enabled 216
  - config queue 215
  - connector data 217
  - connector entry 217
  - connector exit 217
  - job description 217
  - job name 217
  - job queue 217
  - local queue manager 214
  - log file 215
  - node name argument 215
  - options file 216
  - sync queue override 215
- FTF Stage panel
  - all argument 208
  - config file 207
  - config queue 207
  - display version 209
  - FTF transfer id 208
  - local queue manager 207
  - originating queue manager 207
  - purge 207
  - query 207
  - seconds to wait for a reply 208
  - source queue manager 207
  - staged file 208
- FTF Status
  - config file 200
  - config queue 200
  - local queue manager 200
- FTF Status Filter Dialog window 229
- FTF Status Message Client window
  - frames 233
- FTF Status Message Detail window 235
- FTF Status panel
  - destination fullpath filename 203
  - destination queue manager 202
  - display version 203
  - earliest date of transfer 200
  - file containing options 203
  - format 200
  - FTF transfer id 200
  - ignore case sensitivity 203
  - include orphans 200
  - latest date of transfer 201
  - originating queue manager 202
  - purge status queue 200
  - requesting queue manager 202
  - source path 202
  - source queue manager 202
  - status 202
  - user label 200
- FTF transaction identifier
  - AS/400 197
- FTF transaction label 177
- FTF transfer id 182, 200, 208
- FTF version 178
- FTF/MQ Client for 4690 Sender
  - starting 318–319
- FTFCANCL
  - in ISPF panels 160



- FTFCFG
  - arguments 349
  - cfile 349
  - cq 350
  - help 349
  - lqm 349
  - node 349
  - ofile 349
  - syntax 349
  - version 349
- FTFCFGFI logical file name 317
- FTFCFGQ logical file name 317
- FTFDIRMN
  - configuration file 311
  - configuration queue 311
  - instance 312
  - local queue manager 311
  - log file 312
  - nodename 311
  - options file 311
  - rules file 306, 312
  - start 310
  - version 312
- FTFEND 46
  - cfile 47
  - cpt 47
  - cq 47
  - immediate 47
  - in ISPF panels 164
  - ofile 48
  - quiesce 47
  - syntax 46
  - timeout 47
- FTFEND on 4690 331
- FTFExitAuthInfo data structure
  - elements 94
  - function 94
- FTFExitInfo data structure 95, 100
- FTFLOG 356
  - alcunit 359
  - arguments 359
  - blksize 359
  - dirblks 359
  - examples 360
  - ldir 357
  - lqm 357
  - lrecl 359
  - org 358
  - OS/390 arguments 358
  - primary 359
  - q argument 357
  - recfmt 359
  - secondary allocation unit 359
  - secondary argument 359
  - stopping 361
  - syntax 357
  - unit 359
  - unit name 359
  - volser 359
- FTFLogQueue stanza 357
- FTFMGR
  - cfile argument 39
  - cq 39
  - description 38
  - example 39
  - lfile 39
  - ofile 39
  - syntax 38
- FTFMGR on 4690 330
- FTFPING
  - in ISPF panels 161
  - in the GUI 251
  - uses 161
- FTFPING on 4690 330
- FTFPing panel
  - config file 195
  - config queue 195
  - destination queue manager 195
  - display FTF version 196
  - local queue manager 195
  - message size 195
  - options file 196
  - originating queue manager 195
  - priority 196
  - source queue manager 195
  - timeout value 195
- FTFRCV
  - cfile 41

- cq 41
  - description 41
  - example 42
  - lfile 41
  - ofile 41
  - syntax 41
- FTFRCV on 4690 330
- FTFReq function 129
- FTFRVC command
  - syntax 320
- FTFSDR
  - cfile 40
  - cq 40
  - description 39
  - example 40
  - lfile 40
  - ofile 40
  - syntax 39
- FTFSDR command
  - syntax 318– 319
- FTFSTADB 58
- FTFSTART
  - cfile 33
  - command 33
  - cq 33
  - installing authorization exits 97
  - lqm 33, 39– 41, 46
  - ofile 33
  - options files 98
- FTFSTART address space commands 48
  - displaying tasks 49
  - immediately shutting down components 49
  - MODIFY 49
  - PURGE 48
  - shutting down components 48– 49
- FTFSTAT
  - in ISPF 155
  - syntax on AS/400 121
  - using to get AS/400 physical and logical file transfer status 121
- FTFSTATD 58
- FTFSubmitStatusMessage function 84

## G

- GDG dataset name
  - in ISPF file transfer 147
- gradual shutdown 47
- GUI
  - deleting status information 236
  - pinging components 251
  - setting job options 245
  - setting source file options 241
  - setting target file options 243
  - setting user options 247
  - user exits 248
  - viewing detailed status information 235
  - viewing status information 233
- GUI cancel request
  - oqm 256
  - timeout 256
- GUI component shut down
  - component type 258
  - node 258
  - terminate with pending work 258
  - timeout 258
- GUI file transfer
  - allocation unit 245
  - block count 244
  - block size 245
  - configuration file 239
  - data compression 242
  - data persistence 242
  - dqm 239
  - enable cancellation 247
  - exit data 250
  - exit DLL 250
  - exit entry 250
  - exit number 249
  - expiry 246
  - file organization 244
  - file type 242– 243
  - label 247
  - logical records 245
  - make target directory 244
  - message size 246

- oqm 238
- pool name 246
- primary allocation units 245
- priority 246
- record format 245
- reply wait time value 239
- secondary allocation units 245
- source file type 242
- source file type data 242
- sqm 239
- stage persistence 242
- staging 241
- submitting 240
- text wrap 245
- transfer mode 244
- trusted transaction 247
- unit name 245
- volume serial number 245
- wait for reply value 239
- GUI file-transfer request
  - stage only 242
  - Target File Type 244
  - Target File Type Data 244
- GUI ping
  - dqm 252
  - oqm 252–253
  - priority 252
  - sqm 252
  - timeout 252
- GUI stage
  - purge 254
  - purge all 254
  - purge file name 254
  - purge ftfid 254
  - query 254
  - sqm 253
  - wait for reply 254
- GUI status record display
  - altering 234
- GUI status request
  - date range 231
  - dqm 232
  - FTF ID 231
  - label 231

- source file 232
- sqm 231
- status types 232
- target file 232

## H

- help argument
  - FTFCFG 349

## I

- ignore case sensitivity 203
- immediate argument
  - FTFEND 47
- immediate file transfer 182
- immediate shutdown 47
- Immediate Transfer Mode 182
- include orphans 200
- instance argument
  - FTFDIRMN 312
- interfaces 22
- Invalid member name error
  - AS/400 logical file transfer 124
  - AS/400 physical file transfer 124
- Invalid source file error
  - AS/400 logical file transfer 122
  - AS/400 physical file transfer 122
- ISPF
  - reply queue manager 144
- ISPF cancel request
  - FTF ID 161
  - oqm 161
  - timeout 161
- ISPF component shut down
  - component type 164
  - node 164
- ISPF file transfer
  - allocation unit 147
  - AS/400 file type 149

- block size 147
- cancel mode 136
- CCSID 149
- data compression 137
- data persistence 137
- destination path 130
- directory blocks 146
- dqm 130
- Entry Point 151
- exit data 141
- exit DLL 141
- exit entry 141
- exit number 140
- expiry 136
- fail 133
- file ASP 149
- file organization 146
- file text 149
- file type 134
- freplay queue 144
- GDG dataset name 147
- label 130
- library ASP 149
- library text 149
- make target directory 137
- message size 136
- neutral argument option 136
- non-persistent 137
- nopad 134
- off argument option 136
- on argument option 136
- oqm 130
- OS/390 allocation options 145
- pad 134
- persistent 137
- pool name 136
- primary allocation units 147
- priority 136, 162
- record format 147
- record length 147, 149
- record wrap 133
- secondary allocation units 147
- source path 130
- sqm 130

- stage type 136
- staging 137
- transfer mode 133
- truncate 133–134
- trusted transaction 137
- unit name 147
- user exits 138
- volume serial number 147
- wrap 133–134
- ISPF file-transfer request
  - Ddata 152
  - Dtype 152
  - Sdata 151
  - Stype 151
- ISPF notify status
  - notify status
    - in ISPF 144
- ISPF notify type 145
- ISPF panels 127, 155
  - accessing 129
  - cancelling a data-transfer request 160
  - command area 128
  - description 127
  - executing file transfer request 131
  - exiting 165
  - long message area 128
  - pinging components 161
  - requesting a file transfer 129
  - resubmitting a file transfer request 158
  - short message area 128
  - shutting down components 164
  - specifying transfer options 132
  - structure 128
  - Toolbar 128
  - viewing status information 155
- ISPF ping
  - dqm 162
  - message size 162
  - oqm 162
  - sqm 162
  - timeout 162
- ISPF staged request
  - ftf id 154
  - oqm 154

- purge all 154
- purge ftfid 154
- purge source file 154
- query 154
- source path 154
- sqm 154
- wait time 154
- ISPF status information
  - date 158
  - deleting a status record 158
  - displaying log information 158
  - displaying summary information 158
  - event 158
  - queue manager 158
  - status 158
  - time 158
- ISPF status request
  - date range 156
  - dqm 156
  - FTF ID 155
  - label 155
  - oqm 156
  - source file 156
  - sqm 156
  - status types 157
  - target file 156

## J

- job description 213, 217, 221, 223
- job name 213, 217, 221, 223
- Job Options window 245
- job queue 213, 217, 221, 224
- jsdataset argument 91
  - in FTFSTART 91
  - sample 92

## L

- label argument

- file transfer 247
  - in GUI status request 231
  - in ISPF file transfer 130
  - in ISPF status request 155
- latest date of transfer 201
- ldir argument 357
- lfile argument
  - FTFDIRMN 312
  - FTFMGR 39
  - FTFRCV 41
  - FTFSDR 40
- library ASP 185
  - in ISPF file transfer 149
- library text 185
  - in ISPF file transfer 149
- local log file
  - contents 356
  - specifying in a Service 355
  - writing 354
- local queue manager 175, 195, 200, 204, 207, 210, 212, 214, 218, 222
  - AS/400 197
  - FTFCFG 349
  - FTFSTART 33, 39–41, 46
- local queues
  - defining 339–340
  - definition commands 340
- log directory 222, 357
- log file 213, 215, 219, 223
  - FTFMGR 39
  - FTFRCV 41
  - FTFSDR 40
- log information
  - display 159
- log queue 357
  - description 357
  - retrieving information 357
- Logger
  - shutting down 258
- logger queue 222
- logging on to the AS/400 168
- logical filenames
  - 4690 317
- logical record length 359

## N

- logical records
  - in GUI file transfer 245
- long message area
  - ISPF panels 128
- lqm argument 357
  - FTFCFG 349
  - FTFDIRMN 311
  - FTFSTART 33, 39–41, 46
- lrecl argument
  - FTFLOG 359

## M

- make dest directory 178
- make target directory
  - in GUI file transfer 244
  - in ISPF file transfer 137
- Manager
  - arguments 30
  - description 23, 28
  - shutting down 165, 258
  - starting 36
  - submitting status messages 30
- Manager post-process exit 73
- Manager pre-process exit 71
- MaxQueues stanza 341
- menu in AS/400
  - starting 169
- message size 195
  - in GUI file transfer 246
  - in ISPF file transfer 136
  - in ISPF ping 162
- mgr post data 189
- mgr post exit 188
- mgr pre data 188
- mgr pre entry 188
- mgr pre exit 187
- mode of writing file 178
- MQSeries
  - queue capacity 336
- MQSeries client
  - definition 314

- MQSeries object-level security
  - supported by 50
- MVS ALCUNIT 187
- MVS BLKSIZE 187
- MVS GDG model 187
- MVS LRECL 187
- MVS org 186
- MVS primary allocation 187
- MVS RECFMT 187
- MVS secondary allocation 187
- MVS UNIT 187
- MVS VOLSER 187

## N

- neutral argument option
  - in ISPF file transfer 136
- node
  - in GUI component shut down 258
  - in ISPF component shut down 164
- node argument
  - FTFCFG 349
- node name 213, 215, 219
  - FTFCFG 349
- nodename argument
  - FTFDIRMN 311
- non-persistent argument option
  - in ISPF file transfer 137
- nopad argument
  - in ISPF file transfer 134
- noreplace mode 282
- notify type
  - in ISPF 145
- NT service
  - restarting 98
  - starting components manually 35
- number of buffers 147

## O

- OAM Security on MVSOS/390 50
- OAMSecurity
  - enabled in configuration file 50
- off argument option
  - in ISPF file transfer 136
- ofile argument
  - FTFCFG 349
  - FTFDIRMN 311
  - FTFEND 48
  - FTFMGR 39
  - FTFRCV 41
  - FTFSDR 40
  - FTFSTART 33
- on argument option
  - in ISPF file transfer 136
- options file 176, 196, 210, 216, 219, 223
  - AS/400 198
  - description 53
  - example 53
  - structure 53
- oqm argument
  - in GUI cancel request 256
  - in GUI file transfer 238
  - in GUI ping 252–253
  - in ISPF cancel request 161
  - in ISPF file transfer 130
  - in ISPF ping 162
  - in ISPF staged request 154
  - in ISPF status request 156
- org argument
  - FTFLOG 358
- originating queue manager 177, 195, 202, 207
  - AS/400 198
- OS/390
  - allocation options 145
- OS/390 arguments
  - FTF 147
- OS/390 authorization module
  - location 94
  - secondary return codes 95
- Overflow stanza 341

- override max msq size 181
- overrides

- FTFCFG 349
- FTFEND 48
- FTFMGR 39
- FTFRCV 41
- FTFSDR 40
- FTFSTART 33

## P

- pad
  - in ISPF file transfer 134
- pad argument option
  - in ISPF file transfer 134
- pad character 135
- pad records 177
- padding character 177
- partial date values
  - specifying 156, 231
- path to source file 173
- persistent argument option
  - in ISPF file transfer 137
- pinging components
  - in ISPF panels 161
  - in the GUI 251
- polling interval
  - changing 237
- pool design 335
  - determining the number of queues 336
  - setting queue name aliases 337
- pool implementation
  - defining local queues 339–340
  - defining transmission queues 339
- pool name
  - in GUI file transfer 246
  - in ISPF file transfer 136
- Pool name stanza 341
- pool queues
  - resolving names 339
- pools
  - advantages 334

## R

- definition 334
- designing 335
- primary allocation units
  - in GUI file transfer 245
  - in ISPF file transfer 147
- primary argument
  - FTFLOG 359
- Primary Options Menu 130
- priority argument 196
  - in GUI file transfer 246
  - in GUI ping 252
  - in ISPF file transfer 136, 162
- processing message 44– 45
- protected job submission exit 91
- purge 207
- purge all argument
  - in GUI stage 254
  - in ISPF staged request 154
- purge argument
  - in GUI stage 254
- purge file name argument
  - in GUI stage 254
- purge ftfid argument
  - in GUI stage 254
  - in ISPF staged request 154
- purge source file argument
  - in ISPF staged request 154
- purge status queue 200

## Q

- q argument 357
- query 207
- query argument
  - in GUI stage 254
  - in ISPF staged request 154
- queue manager
  - in ISPF status information 158
- queue manager aliases 337
  - definition commands 340
- queue name aliases
  - requirements for 337

- queue names
  - resolving 339
- QueueName stanza 341
- quiesce argument
  - FTFEND 47

## R

- RCDFMT
  - specifying a source portal 151– 152, 174– 175, 242, 244
- Receiver
  - description 24, 31
  - shutting down 165, 258
- receiver portal data 193
- receiver portal entry 193
- receiver portal exit 193
- receiver post data 192
- receiver post entry 191
- receiver post exit 191
- Receiver post-process exit 82
- receiver pre data 191
- receiver pre entry 191
- receiver pre exit 190
- Receiver pre-process exit 80
- recfmt argument
  - FTFLOG 359
- record format 359
  - in GUI file transfer 245
  - in ISPF file transfer 147
- record length 185
  - in ISPF file transfer 147, 149
- record wrap
  - in ISPF file transfer 133
- record wrapping 178
- reply queue
  - in ISPF 144
- reply queue manager
  - in ISPF 144
- reply wait time value
  - in GUI file transfer 239
- request canceled message 43– 45



- request completed message 43, 45
- request expired message 43–45
- request failed message 43–45
- request flagged for cancel 44
- request module
  - definition 29
  - list 29
- request received message 44–45
- request recovering message 45
- request submitted message 43
- request transmitted message 44
- requesting queue manager 202
- resubmitting a file-transfer request
  - in ISPF panels 158
- return codes
  - in authorization modules 95
- rfile argument
  - FTFDIRMN 312
- rules file
  - FTFDIRMN 306

## S

- Sample Exit on OS/400 63
- Scenario One 55
- Scenario Two 56
- scheduling a one time task
  - one time task 299
- scheduling a repeating task
  - repeating task 290
- Sdata
  - in ISPF file-transfer request 151
- sdata parameter
  - syntax 109
- secondary allocation units
  - in GUI file transfer 245
  - in ISPF file transfer 147
- secondary argument
  - FTFLOG 359
- secondary return codes
  - in OS/390 authorization module 95
  - setting in authorization modules 95

- seconds to wait for reply
  - FTF Request 178
  - FTF Stage 208
- seconds to wait for response
  - FTF Stage 205
- security
  - supported by 50
- selecting FTF commands 170
- Send from Stage argument
  - in stage options 143
- Sender
  - description 24, 31
  - processing summary 31
  - shutting down 165, 258
  - starting 37
- sender portal data 192
- sender portal entry 192
- sender portal exit 192
- sender post data 190
- sender post entry 190
- sender post exit 190
- Sender post-process exit 78
- sender pre data 189
- sender pre entry 189
- sender pre exit 189
- Sender pre-process exit 76
- service component 34
- setting environment variables 351
- setting job options 245
- setting source file options 241
- setting target file options 243
- setting user options 247
- short message area
  - ISPF panels 128
- Shut Down Components panel 164
- shutdown
  - gradual 47
  - immediate 47
- shutting down components
  - in ISPF panels 164
- source data
  - specifying 109
- source file
  - in GUI status request 232

- in ISPF status request 156
- source file data 173
- Source file is not a physical or logical file error
  - AS/400 logical file transfer 123
  - AS/400 physical file transfer 123
- source file type 173
  - in GUI file transfer 242
  - specifying 109
- source file type data
  - in GUI file transfer 242
- source fullpath filename 202
- source path
  - in ISPF file transfer 130
- source path argument
  - in ISPF staged request 154
  - in stage options 142
- source path file name 173
- source queue manager 195, 202, 207
- source queue manager name 173
- specifying a log directory 357
- specifying a log queue 357
- specifying directory blocks 359
- specifying partial date value 156, 231
- specifying the staging queue 143, 181
- sqm argument
  - in GUI file transfer 239
  - in GUI ping 252
  - in GUI stage 253
  - in GUI status request 231
  - in ISPF file transfer 130
  - in ISPF ping 162
  - in ISPF staged request 154
  - in ISPF status request 156
- stage only 181
  - in GUI file-transfer request 242
- Stage Only argument
  - in stage options 143
- stage options
  - Send from stage 143
  - setting 141
  - source path 142
  - Stage Only 143
  - Staged FTF Identifier 142
- stage persistent 179
- stage persistence
  - in GUI file transfer 242
- stage the source 181
- stage type
  - in ISPF file transfer 136
- staged file 208
- Staged FTF Identifier argument
  - in stage options 142
- staging
  - in GUI file transfer 241
  - in ISPF file transfer 137
- staging completed 44
- staging queue
  - specifying 143, 181
- Starting an AS/400 Session 36
- starting an eNDI Client for 4690 Receiver 320
- starting an eNDI Client for 4690 Sender 318–319
- starting components 36
  - all at once 33–34
  - individually 38
- Starting Components Individually 38
  - OS/390 34
  - Windows NT 35
- Starting FTF in UNIX 38
- Starting FTF on AS/400 36
- Starting FTF on OS/390 Systems 33
- Starting the FTF Logger 38
- Starting the FTF Manager 36, 38
- Starting the FTF Receiver 37, 41
- Starting the FTF Sender 37, 39
- status 202
  - in ISPF status information 158
- Status Daemon
  - shutting down 258
- Status Filter panel 155
- status information
  - filtering 229
  - viewing 155, 229
- Status List panel 157
- Status Messages frame 234
- status offload daemon 57
  - command line definitions 63
  - configuration file definitions 61
  - data flow 59

- database schema 65
- overview 58
- relational database 58
- retrieving from sample database 65
- sample 58
- starting 60
- stopping 60
- status subsystem
  - description 24
- status types
  - in GUI status request 232
  - in ISPF status request 157
- STRFTFMGR panel
  - job queue 213
- STRFTFSDR panel
  - config file 214
- Stype
  - in ISPF file-transfer request 151
- summary information
  - display 159
  - ISPF status information 158
- sync queue override 215, 218

## T

- target file
  - in GUI status request 232
  - in ISPF status request 156
- target file block size 359
- Target file is not a physical or logical file error
  - AS/400 logical file transfer 124
  - AS/400 physical file transfer 124
- Target file is not compatible with the source file
  - AS/400 logical file transfer 125
  - AS/400 physical file transfer 125
- Target File Options window 243
- target file organization 358
- Target File Type
  - GUI file-transfer request 244
- Target File Type Data
  - GUI file-transfer request 244
- target file unit name 359
- target volume serial number 359
- TBL
  - specifying a source portal 151– 152, 173, 175, 242, 244
- terminate with pending work 205
  - in GUI component shut down 258
- text wrap
  - in GUI file transfer 245
- The FTF Manager 28
- The FTF Manager and the Status Subsystem 43
- The FTF Receiver 31
- The FTF Sender 31
- The MODIFY Command 49
- The PURGE Command 48
- time
  - in ISPF status information 158
- timeout
  - in GUI component shut down 258
  - in GUI ping 252
- timeout argument
  - FTFEND 47
  - in GUI cancel request 256
  - in ISPF cancel request 161
  - in ISPF ping 162
- timeout value 195
- Tivoli Customer Support 17
- Tivoli Data Exchange
  - description 13– 14
- Toolbar
  - ISPF panels 128
- transfer mode
  - in GUI file transfer 244
  - in ISPF file transfer 133
- Transfer Options panel 132
- transfer priority 181
- transmission persistent 179
- transmission queues
  - defining 339
  - definition commands 339
- truncate argument option
  - in ISPF file transfer 133– 134
- trusted transaction
  - in GUI data transfer 247
  - in ISPF file transfer 137

## X

trusted transfer 179

## U

Unable to validate source file

AS/400 logical file transfer 122

AS/400 physical file transfer 122

Unable to validate target file

AS/400 logical file transfer 124

AS/400 physical file transfer 124

unit argument

FTFLOG 359

unit name

in GUI file transfer 245

in ISPF file transfer 147

user exit

4690 329

user exits

adding in GUI 248

compiling and linking DLLs 87

definition 68

examples 86

in GUI file transfer 248

in ISPF file transfer 138

lists 68

Manager post-process exit 73

Manager pre-process 71

overview 70

Receiver post-process exit 82

Receiver pre-process exit 80

removing in GUI 251

Sender post-process exit 78

Sender pre-process exit 76

user label argument 200

Using FTFSTART Address Space Commands 48

Using the AS/400 FTF Menu 36

Using the FTFEND Command 46

using the menu on AS/400 36

## V

version argument

FTFCFG 349

FTFDIRMN 312

viewing detailed status information

in the GUI 235

viewing status information

in the GUI 233

volser argument

FTFLOG 359

volume serial number 359

in GUI file transfer 245

in ISPF file transfer 147

## W

wait for reply argument

in GUI stage 254

wait for reply value

in GUI file transfer 239

wait time argument

in ISPF staged request 154

Win 32 60

wrap argument option

in ISPF file transfer 133–134

## X

xmit queues (see transmission queues)