Oracle® Application Server Integration InterConnect

Adapter for FTP Installation and User's Guide

10*g* Release 2 (10.1.2)

B14073-02

December 2005

Oracle Application Server Integration InterConnect Adapter for FTP Installation and User's Guide, 10*g* Release 2 (10.1.2)

B14073-02

Primary Author: Rima Dave

Contributing Author: Vimmy Raj

Contributor: Sivaraj Subbaiyan, Maneesh Joshi, Rahul Pathak, Harish Sriramulu

Contents

A Frequently Asked Questions

B Example of the adapter.ini File

Index

Preface

This Preface contains these topics:

- Audience
- Documentation Accessibility
- Related Documents
- Conventions

Audience

Oracle Application Server Integration InterConnect Adapter for FTP Installation and User's Guide is intended for system administrators of OracleAS Integration InterConnect who perform the following tasks:

- install applications
- maintain applications

To use this document, you need to know how to install and configure OracleAS Integration InterConnect.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at

`http://www.oracle.com/accessibility/`

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Related Documents

For more information, refer to these Oracle resources:

- *Oracle Application Server Integration InterConnect User's Guide*
- *Oracle Application Server Integration InterConnect Installation Guide*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Introduction

This chapter provides an overview on how to use Oracle Application Server Integration InterConnect (OracleAS Integration InterConnect) Adapter for File Transfer Protocol (FTP adapter). It contains the following topics:

- FTP Adapter Overview
- FTP Adapter System Requirements
- Known FTP Adapter Limitations

1.1 FTP Adapter Overview

The FTP adapter enables an FTP application to be integrated with other applications using OracleAS Integration InterConnect. The FTP adapter is useful in all Enterprise Application Integration (EAI) environments that use FTP. EAI is the integration of applications and business processes within the same company.

The FTP adapter can monitor incoming messages which are in the form of FTP files placed in a remote FTP server or on local file systems. The FTP adapter is also capable of sending messages to remote FTP servers by proxy host. The payload for this adapter can be XML data or D3L data.

> **See Also:** *Oracle Application Server Integration InterConnect User's Guide*

Figure 1–1 shows the data flow of incoming messages, from remote FTP server/local file system to OracleAS Integration InterConnect.

Figure 1–1 Incoming Messages Data Flow

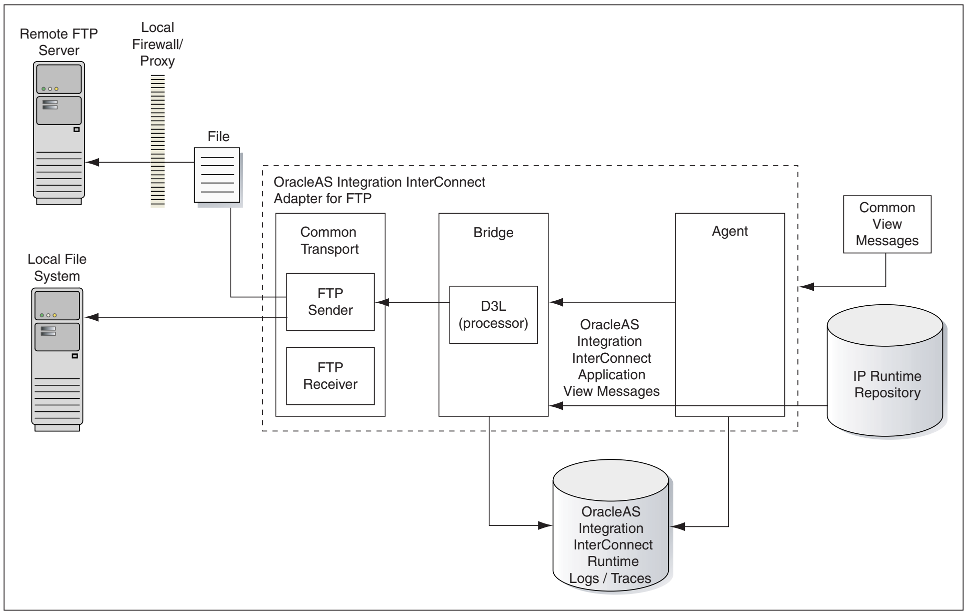
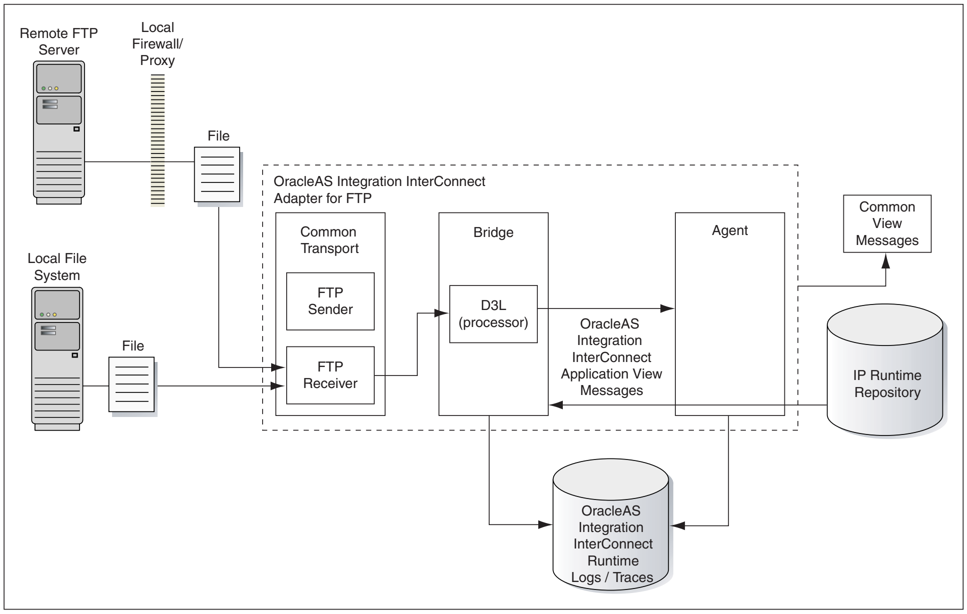


Figure 1–2 shows the data flow of outgoing messages from OracleAS Integration InterConnect to the remote FTP Server.

Figure 1–2 Outgoing Messages Data Flow



1.2 FTP Adapter System Requirements

The following sections describe the FTP adapter system requirements:

- Hardware Requirements
- Software Requirements

1.2.1 Hardware Requirements

Table 1–1 lists the hardware requirements for installing the FTP adapter.

Table 1–1 Hardware Requirements

Hardware	Windows	UNIX
Disk Space	400 MB	400 MB
Memory	512 MB	512 MB

1.2.2 Software Requirements

The following sections describe the FTP adapter software requirements:

- Operating System Requirements
- JRE Requirements

Operating System Requirements

Table 1–2 lists the operating system requirements for installing the FTP adapter.

Table 1–2 Operating System Requirements

Operating System	Version
HP Tru64	HP Tru64 UNIX (Alpha) 5.1b
HP-UX	HP-UX (PA-RISC) 11.11, 11.23
IBM AIX	AIX (POWER) version 5.2
Linux (x86)	Red Hat Enterprise Linux 2.1, 3.0 SuSE SLES8, SLES9
Sun SPARC Solaris	Sun SPARC Solaris 2.8 and 2.9
Microsoft Windows	Windows XP Professional, Windows 2000 (SP3 or higher)

JRE Requirements

OracleAS Integration InterConnect uses Java Runtime Environment (JRE) 1.4, which is installed with its components.

1.3 Known FTP Adapter Limitations

The FTP adapter has the following limitations:

- Does not support secure FTP
- Does not support file-filtering features
- Can receive only from a single FTP or file endpoint
- Restricts the sending endpoint and receiving endpoint to FTP and file endpoints

2 Installation and Configuration

This chapter describes how to install and configure the FTP adapter. It contains the following topics:

- Installing the FTP Adapter
- Installing Multiple FTP Adapters in the Same Oracle Home
- Configuring the FTP Adapter
- Uninstalling the FTP Adapter

2.1 Installing the FTP Adapter

The FTP adapter must be installed in an existing Oracle home Middle Tier for OracleAS Integration InterConnect 10*g* Release 2 (10.1.2).

This section contains the following topics:

- Preinstallation Tasks
- Installation Tasks
- Postinstallation Tasks

2.1.1 Preinstallation Tasks

Refer to the following guides before installing the FTP adapter:

- *Oracle Application Server Installation Guide*for information about Oracle Universal Installer startup.
- *Oracle Application Server Integration InterConnect Installation Guide* for information about software, hardware, and system requirements for OracleAS Integration InterConnect.

> **Note:** OracleAS Integration InterConnect Hub is installable through the OracleAS Integration InterConnect Hub installation type. You must install the OracleAS Integration InterConnect Hub before proceeding with the FTP adapter installation.

2.1.2 Installation Tasks

To install the FTP adapter:

1. In the Available Product Components page of the OracleAS Integration InterConnect installation, select **OracleAS Integration InterConnect Adapter for FTP 10.1.2.0.2** and click **Next**.
2. The Set Oracle Wallet Password page is displayed. Enter and confirm the password, which will be used to manage the OracleAS Integration InterConnect installation. Click **Next**.
   - Go to step 3 if installing the FTP adapter in an OracleAS Middle Tier Oracle home that does not have an InterConnect component already installed. Ensure that the OracleAS Integration InterConnect hub has been installed.
   - Go to step 4 if installing the FTP adapter in an OracleAS Middle Tier Oracle home that has an existing InterConnect component. Ensure that it is a home directory to an OracleAS Integration InterConnect component.

   Note: All passwords are stored in Oracle Wallet. Refer to Section A.7, "How do I secure my passwords?" in Appendix A, "Frequently Asked Questions" for more details on how to modify and retrieve the password by using Oracle Wallet.

3. The Specify Hub Database Connection page is displayed. Enter information in the following fields:
   - Host Name: The host name of the computer where the hub database is installed.
   - Port Number: The TNS listener port for the hub database.
   - Database SID: The System Identifier (SID) for the hub database.
   - Password: The password for the hub database user.
4. Click **Next**. The Specify FTP adapter Name page is displayed.
5. Enter the application to be defined. Blank spaces are not permitted. The default value is `myFTPApp`.

   Note: You can change the application name in iStudio after installation. In such case, you need to specify the password corresponding to new application name in Oracle Wallet.

   For more informtion, refer to the following sections in Appendix A, "Frequently Asked Questions":

   - Section A.4, "The FTP adapter is not starting. What could be the reason?"
   - Section A.7, "How do I secure my passwords?"

6. Click **Next**. The Specify FTP adapter Usage page is displayed.
7. Select one of the following options and go to the step specified.

If You Select...	Then Click Next and Go to Step...
Configure for both sending and receiving messages	8
Configure for sending messages ONLY	8

If You Select...	Then Click Next and Go to Step...
Configure for receiving messages ONLY	10

Note: You can change the values for these selections later by editing the parameter settings in the `adapter.ini` file.

8. Enter the following information in the Configure Sending Endpoint Information page:

   - Username: The user name for the FTP server.
   - Password: The user password for the FTP server.
   - FTP Mode: The mode of access used to send information to the specified URL. Select either binary or ASCII.
   - URL: The URL to be used for sending information. Enter one of the following:
     - For sending to an FTP server: *`ftp://host name/path`*
     - For sending to a local file system: *`file://localhost/path`*

   Note: If the sender endpoint is a local file system, then the user name, password, and file type are not required.

9. Click **Next**. The installation page that is displayed next is based on the selection you made in Step 7.

If You Selected...	Then Go to Step...
Configure for both sending and receiving messages	10
Configure for sending messages ONLY	11

10. Enter the following information in the Configure Receiving Endpoint Information page:

    - Username: The user name account of the FTP server from which the Oracle Application Server Integration InterConnect Hub receives messages.
    - Password: The password for the user name account.
    - FTP Mode: The mode of access used to receive information from the specified URL. Select either binary or ASCII.
    - URL: The FTP URL to be used for receiving information. Enter one of the following:
      - For sending to an FTP server: *`ftp://host name/path`*
      - For sending to a local file system: *`file://localhost/path`*

    Note: If the sender endpoint is a local file system, then the user name, password, and file type are not required.

Caution: During run time, the FTP adapter connects to the FTP server or accesses the file system and removes the files in the directory specified by the receiving endpoint after processing. Oracle recommends that you create a dedicated FTP account or user account (if the local file system is used for the receiving endpoint) for testing and deploying this adapter.

11. Click **Next**. The Summary page is displayed.
12. Click **Install** to install the FTP adapter. The following table lists the platform and the directory in which the FTP adapter will be installed.:

Platform	Directory
UNIX	*ORACLE_HOME*/integration/interconnect/adapters/*Application*
Windows	*ORACLE_HOME*\integration\interconnect\adapters\Application

You defined the value of `Application` in Step 5.

13. Click **Exit** on the End of Installation page to exit the FTP adapter installation.

2.1.3 Postinstallation Tasks

The FTP adapter installation creates the `adapter.ini` file that consists of configuration parameters read by the FTP adapter at startup. These configuration parameter settings are suitable for most FTP application environments. To customize the `adapter.ini` file parameter settings for the FTP application, refer to the following sections:

- Customizing the Payload Data Type
- Customizing the Sending Endpoints
- Customizing the Receiving Endpoints

2.1.3.1 Customizing the Payload Data Type

Payload data is the data sent between applications. To change the payload data type from the default of XML to the Data Definition Description Language (D3L), edit the following parameters in the `adapter.ini` file.

1. Set the `ota.type` parameter to the payload type D3L.

```
ota.type=D3L
```

2. Copy the D3L XML files associated with the FTP application to the directory in which the `adapter.ini` file is located.
3. Set the `ota.d3ls` parameter to specify the D3L files associated with the FTP application.

```
ota.d3ls=person1.xml,person2.xml
```

See Also: `ota.type` and `ota.d3ls` parameter descriptions in Table 2–9

2.1.3.2 Customizing the Sending Endpoints

To customize the sending endpoints (destinations) for messages, edit the following parameters in the `adapter.ini` file.

1. Edit the `ota.send.endpoint` parameter, or leave it blank if it acts only as a receiver. For example:

   - For a remote file system. `ota.send.endpoint=ftp://foo.com/test`
   - For a local file system. `ota.send.endpoint=file://localhost/test`

   If the endpoint is a local file system, then leave the following parameters blank:

   - `file.sender.file_type`
   - `file.sender.password`
   - `file.sender.proxy_host`
   - `file.sender.proxy_port`

2. Set the `file.sender.file_type` parameter to the file type used in FTP. For example:

   ```
   file.sender.file_type=BINARY
   ```

3. Update the `file.sender.user` and `file.sender.password` parameters with the information of the FTP account that serves as the sending endpoint.

4. If a proxy host is needed, then enter the values for the `file.sender.proxy_host` and `file.sender.proxy_port` parameters.

5. Set the `file.sender.staging_dir` parameter. This parameter prevents partial files from being picked up by external applications.

6. Set the `file.sender.file_name_rule` parameter. This parameter controls how the adapter generates the file name.

7. If you need to modify the contents of an outgoing message before it is sent by the transport layer, then you can customize it by implementing the `FileSenderCustomizer` interface. You need to set the `file.sender.customize_class` parameter to the name of the customizing class.

   See Also: Section 3.2.2, "FTP Sender"

2.1.3.3 Customizing the Receiving Endpoints

To customize the receiving FTP or file endpoints for messages, edit the following parameters in the `adapter.ini` file.

1. Edit the `ota.receive.endpoint` parameter, or leave it blank if the adapter acts only as a sender. For example:

   - For a remote file system: `ota.receive.endpoint=ftp://foo.com/test`
   - For a local file system: `ota.receive.endpoint=ftp://localhost/test`

   If the endpoint is a local file system, then leave the following parameter blank:

   - `file.receiver.file_type`
   - `file.receiver.password`
   - `file.receiver.proxy_host`
   - `file.receiver.proxy_port`

Warning: Do not set the ota.receive.endpoint parameter to a personal file directory because the files in that directory will be consumed and deleted by the FTP adapter after processing.

---

2. Set the `file.receiver.file_type` parameter to the file type used in FTP. For example:

   ```
   file.receiver.file_type=BINARY
   ```

3. Update the `file.receiver.user` and `file.receiver.password` parameters with the information of the FTP account that serves as the receiving endpoint.
4. If a proxy host is needed, then enter the required values for the `file.receiver.proxy_host` and `file.receiver.proxy_port` parameters.
5. Set the `file.exception.exception_dir` parameter to a local file system directory that stores files. For example:

   ```
   file.receiver.exception_dir=/tmp/error
   ```

6. Set the `file.receiver.polling_interval` parameter to the time interval in milliseconds to poll the FTP server or local file system. For example:

   ```
   file.receiver.polling_interval=2000
   ```

7. Set the `file.receiver.max_msgs_retrieved` parameter to the maximum number of messages to retrieve in polling a session. For example:

   ```
   file.receiver.max_msgs_retrieved=10
   ```

8. If you need to modify the contents of an incoming message before it is processed by the bridge, such as to remove an extra line in a file, then you customize it by implementing the `FileSenderCustomizer` interface. You need to set the `file.receiver.customize_class` parameter to the name of the customizing class.

2.2 Installing Multiple FTP Adapters in the Same Oracle Home

To install multiple instances of the FTP adapter in same Oracle home, use the `copyAdapter` script located in the *`ORACLE_HOME`*`/integration/interconnect/bin` directory.

Usage: `copyAdapter` *`app1 app2`*

For example, you have one instance of FTP adapter with name `myFTPApp` installed on a computer. To install another instance of the FTP adapter with name `myFTPApp1` in the same Oracle home, use the following command:

```
copyAdapter myFTPApp myFTPApp1
```

The `copyAdapter` script is copied to the following `bin` directory only during Hub installation:

- UNIX: `ORACLE_HOME/integration/interconnect/bin`
- Windows: `ORACLE_HOME\integration\interconnect\bin`

If you need to use this script to create multiple adapters on a spoke computer, then copy the script to the `bin` directory on the spoke computer, and edit the script to reflect the new Oracle home.

After running the `copyAdapter` script, If you want to manage or monitor the newly installed adapter through Oracle Enterprise Manager 10*g* Application Server Control Console, then you need to modify the `opmn.xml` file by adding information about the new instance. For example, you have created a new instance of the FTP adapter `myFTPApp1` by using the `copyAdapter` script. To manage the `myFTPApp1` adapter through Enterprise Manager, perform the following:

1. Navigate to the *`MiddleTier`*`\bin` directory and run the following command to stop the Enterprise Manager:

   ```
   emctl stop iasconsole
   ```

2. Next, specify the information about this new instance in the `opmn.xml` file located in the *`ORACLEMIDDLETIER_HOME`*`/opmn/conf` directory as follows:

   ```
   <process-type id="myFTPApp1" module-id="adapter" working-dir="$ORACLE_
   HOME/integration/interconnect/adapters/myFTPApp1" status="enabled">
          <start timeout="600" retry="2"/>
          <stop timeout="120"/>
          <port id="icadapter_dmsport_range" range="15701-15800"/>
          <process-set id="myFTPApp1" restart-on-death="true" numprocs="1">
               <module-data>
                   <category id="start-parameters">
                       <data id="java-parameters" value="-Xms8M"/>
                       <data id="class-name"
                        value="oracle.oai.agent.service.AgentService"/>
                   </category>
                   <category id="stop-parameters">
                       <data id="java-parameters" value="-mx64m"/>
                       <data id="class-name"
                        value="oracle.oai.agent.proxy.ShutdownAgent"/>
                       <data id="application-parameters"
                        value="persistence/Agent.ior"/>
                   </category>
               </module-data>
          </process-set>
   </process-type>
   ```

   The `opmn.xml` file would appear like this:

   ```
   <process-type id="myFTPApp" module-id="adapter" working-dir="$ORACLE
   _HOME/integration/interconnect/adapters/myFTPApp" status="enabled">
          <start timeout="600" retry="2"/>
          <stop timeout="120"/>
          <port id="icadapter_dmsport_range" range="15701-15800"/>
          <process-set id="myFTPApp" restart-on-death="true" numprocs="1">
               <module-data>
                   <category id="start-parameters">
                       <data id="java-parameters" value="-Xms8M"/>
                       <data id="class-name"
                        value="oracle.oai.agent.service.AgentService"/>
                   </category>
                   <category id="stop-parameters">
                       <data id="java-parameters" value="-mx64m"/>
                       <data id="class-name"
                value="oracle.oai.agent.proxy.ShutdownAgent"/>
                <data id="application-parameters"
                 value="persistence/Agent.ior"/>
            </category>
        </module-data>
     </process-set>
   ```

```
</process-type>

<process-type id="myFTPApp1" module-id="adapter" working-dir="$ORACLE
_HOME/integration/interconnect/adapters/myFTPApp1" status="enabled">
       <start timeout="600" retry="2"/>
       <stop timeout="120"/>
       <port id="icadapter_dmsport_range" range="15701-15800"/>
       <process-set id="myFTPApp1" restart-on-death="true" numprocs="1">
        <module-data>
          <category id="start-parameters">
              <data id="java-parameters" value="-Xms8M"/>
              <data id="class-name"
               value="oracle.oai.agent.service.AgentService"/>
          </category>
          <category id="stop-parameters">
               <data id="java-parameters" value="-mx64m"/>
               <data id="class-name"
                value="oracle.oai.agent.proxy.ShutdownAgent"/>
               <data id="application-parameters"
                value="persistence/Agent.ior"/>
          </category>
        </module-data>
       </process-set>
</process-type>
```

3. Save the `opmn.xml` file.
4. Navigate to the `*MiddleTier*\opmn\bin` directory and run the following command to reload the OPMN:

   ```
   opmnctl reload
   ```

5. You can start the `myFTPApp1` adapter by using the following command

   ```
   opmnctl startproc ias-component="InterConnect" process-type="myFTPApp1"
   ```

6. Navigate to the `*MiddleTier*\bin` directory and run the following command to start the Enterprise Manager:

   ```
   emctl start iasconsole
   ```

7. Login to the Oracle Enterprise Manager 10*g* Application Server Control Console to view and manage the newly installed or copied adapter. For information about how to use Oracle Enterprise Manager 10*g* Application Server Control Console , refer to the *Oracle Application Server Integration InterConnect User's Guide*

> **Note:** While installing multiple adapters in the same computer, the copyadapter script does not create entries for the new adapter's password in the Oracle Wallet. You need to manually create a password for this new adapter using the Oracle Wallet Manager. To store the password in Oracle Wallet, use the following format:
>
> `ApplicationName/password`
>
> The number of entries is dependent on the type of adapter. For example, Database Adapter needs two entries whereas AQ Adapter needs only one entry. For more information about how to manage your passwords in Oracle Wallet, refer to Section A.7, "How do I secure my passwords?"in Appendix A, "Frequently Asked Questions"

2.3 Configuring the FTP Adapter

You can configure the FTP adapter after installation. The following tables describe the location and details of the configuration files.

Table 2–1 describes the location where the adapter is installed.

Table 2–1 FTP Adapter Directory

Platform	Directory
UNIX	*ORACLE_HOME*/integration/interconnect/adapters/*Application*
Windows	*ORACLE_HOME*\integration\interconnect\adapters\Application

Table 2–2 describes the executable files of the FTP adapter.

Table 2–2 FTP Executable Files

File	Description
`start` (UNIX)	Does not use parameters; starts the adapter.
`start.bat` (Windows)	Does not use parameters; starts the adapter.
`stop` (UNIX)	Does not use parameters; stops the adapter.
`stop.bat` (Windows)	Does not use parameters; stops the adapter.

Table 2–3 describes the FTP adapter configuration files.

Table 2–3 FTP Configuration Files

File	Description
`adapter.ini` (UNIX)	Consists of all the initialization parameters that the adapter reads at startup.
`adapter.ini` (Windows)	Consists of all the initialization parameters that the adapter reads at startup.

Table 2–4 describes the directories used by the FTP adapter.

Table 2–4 FTP Directories

Directory	Description
`logs`	The adapter activity is logged in subdirectories of the `logs` directory. Each time the adapter is run, a new subdirectory is created for the `log.xml` log file.
`persistence`	The messages are made available in this directory. Do not edit this directory or its files.

2.3.1 Using the Application Parameter

Adapters do not have integration logic. The FTP adapter has a generic transformation engine that uses metadata from the repository as run-time instructions to perform transformations. The application parameter defines the capabilities of an adapter, such as the messages to be published and subscribed, and the transformations to be performed. The application parameter allows the adapter to retrieve only the relevant metadata from the repository. The application parameter must match the corresponding application name that will be defined in iStudio under the Applications folder.

If you use prepackaged metadata, then import it into the repository and start iStudio to find the corresponding application under the Applications folder. You can use this as the application name for the adapter you are installing.

> **See Also:** Step 4 on page 2-2

2.3.2 FTP Adapter .ini File Settings

The following `.ini` files are used to configure the FTP adapter:

- hub.ini Files
- adapter.ini Files

2.3.2.1 hub.ini Files

The FTP adapter connects to the hub database using parameters in the `hub.ini` file located in the `hub` directory. Table 2–5 gives a description and an example for each parameter.

Table 2–5 hub.ini Parameters

Parameter	Description	Example
`hub_host`	The name of the computer hosting the hub database. There is no default value. The value is set during installation.	`hub_host=mpscottpc`
`hub_instance`	The SID of the hub database. There is no default value. The value is set during installation.	`hub_instance=orcl`
`hub_port`	The TNS listener port number for the hub database instance. There is no default value. The value is set during installation.	`hub_port=1521`
`hub_username`	The name of the hub database schema (or user name). The default value is `ichub`.	`hub_username=ichub`
`repository_name`	The name of the repository that communicates with the adapter. The default value is `InterConnectRepository`.	`repository_name=InterConnectRepository`

Oracle Real Application Clusters hub.ini Parameters

When a hub is installed on an Oracle Real Application Clusters database, the parameters listed in Table 2–6 represent information about additional nodes used for connection and configuration. These parameters are in addition to the default parameters for the primary node. In Table 2–6, `x` represents the node number which can range from 2 to the total number of nodes in cluster. For example, if the cluster setup contains 4 nodes, then `x` can be a value between 2 and 4.

Table 2–6 Oracle Real Application Clusters hub.ini Parameters

Parameter	Description	Example
`hub_host`*`x`*	The host where the Real Application Clusters database is installed.	`hub_host2=dscott13`
`hub_instance`*`x`*	The instance on the respective node.	`hub_instance2=orcl2`
`hub_num_nodes`	The number of nodes in a cluster.	`hub_num_nodes=4`
`hub_port`*`x`*	The port where the TNS listener is listening.	`hub_port2=1521`

2.3.2.2 adapter.ini Files

The agent component of the FTP adapter reads the adapter.ini file at run time to access FTP adapter parameter configuration information. Table 2–7 gives a description and an example for each parameter.

Table 2–7 adapter.ini Parameters

Parameter	Description	Example
agent_admin_port	Specifies the port through which the adapter can be accessed through firewalls. Possible value: A valid port number Default value: None	agent_admin_port=1059
agent_delete_file_cache_at_startup	Specifies whether to delete the cached metadata during startup. If any agent caching method is enabled, then metadata from the repository is cached locally on the file system. Set the parameter to true to delete all cached metadata on startup. Possible values: true or false Default value: false **Note:** After changing metadata or DVM tables for the adapter in iStudio, you must delete the cache to guarantee access to new metadata or table information.	agent_delete_file_cache_at_startup=false
agent_dvm_table_caching	Specifies the Domain Value Mapping (DVM) table caching algorithm. Possible values: ■ startup: Cache all DVM tables at startup. This may be time-consuming if there are many tables in the repository. ■ demand: Cache tables as they are used. ■ none: No caching. This slows down performance. Default value: demand	agent_dvm_table_caching=demand
agent_log_level	Specifies the amount of logging necessary. Possible values: 0=errors only 1=status and errors 2=trace, status, and errors Default value: 1	agent_log_level=2
agent_lookup_table_caching	Specifies the lookup table caching algorithm. Possible values: ■ startup: Cache all lookup tables at startup. This may be time-consuming if there are many tables in the repository. ■ demand: Cache tables as they are used. ■ none: No caching. This slows down performance. Default value: demand	agent_lookup_table_caching=demand
agent_max_ao_cache_size	Specifies the maximum number of application object metadata to cache. Possible value: An integer greater than or equal to 1. Default value: 200	agent_max_ao_cache_size=200

Table 2–7 (Cont.) adapter.ini Parameters

Parameter	Description	Example
agent_max_co_cache_size	Specifies the maximum number of common object metadata to cache. Possible value: An integer greater than or equal to 1. Default value: 100	agent_max_co_cache_size=100
agent_max_dvm_table_cache_size	Specifies the maximum number of DVM tables to cache. Possible value: An integer greater than or equal to 1. Default value: 200	agent_max_dvm_table_cache_size=200
agent_max_lookup_table_cache_size	Specifies the maximum number of lookup tables to cache. Possible value: Any integer greater than or equal to 1. Default value: 200	agent_max_lookup_table_cache_size=200
agent_max_message_metadata_cache_size	Specifies the maximum number of message metadata (publish/subscribe and invoke/implement) to cache. Possible value: An integer greater than or equal to 1. Default value: 200	agent_max_message_metadata_cache_size=200
agent_max_queue_size	Specifies the maximum size to which internal OracleAS Integration InterConnect message queues can grow. Possible value: An integer greater than or equal to 1. Default value: 1000	agent_max_queue_size=1000
agent_message_selector	Specifies conditions for message selection when the adapter registers its subscription with the hub. Possible value: A valid Oracle Advanced Queue message selector string (such as'%,aqapp,%'). Default value: None	agent_message_selector=%,aqapp,%
agent_metadata_caching	Specifies the metadata caching algorithm. Possible values: ▪ startup: Cache everything at startup. This may be time-consuming if there are many tables in the repository. ▪ demand: Cache metadata as it is used. ▪ none: No caching. This slows down performance. Default value: demand	agent_metadata_caching=demand
agent_persistence_cleanup_interval	Specifies how often to run the persistence cleaner thread in milliseconds. Possible value: An integer greater than or equal to 30000 milliseconds. Default value: 60000	agent_persistence_cleanup_interval=60000
agent_persistence_queue_size	Specifies the maximum size of internal OracleAS Integration InterConnect persistence queues. Possible value: An integer greater than or equal to 1. Default value: 1000	agent_persistence_queue_size=1000

Table 2–7 (Cont.) adapter.ini Parameters

Parameter	Description	Example
agent_persistence_retry_interval	Specifies how often the persistence thread retries when it fails to send an OracleAS Integration InterConnect message. Possible value: An integer greater than or equal to 5000 milliseconds. Default value: 60000	agent_persistence_retry_interval=60000
agent_pipeline_from_hub	Specifies whether to set the pipeline for messages from the hub to the bridge. If you set the pipeline to false, then the file persistence is not used in that direction. Possible value: true, false Default value: false	agent_pipeline_from_hub=false
agent_pipeline_to_hub	Specifies whether to set the pipeline for messages from the bridge to the hub. If you set the pipeline to false, then the file persistence is not used in that direction. Possible value: true, false Default value: false	agent_pipeline_to_hub=false
agent_reply_message_selector	Specifies the application instance to which the reply must be sent. This parameter is used if multiple adapter instances exist for the given application and given partition. Possible value: A string built using the application name (parameter:application) concatenated with the instance number (parameter:instance_number) Default value: None	If application=aqapp, instance_number=2,then agent_reply_message_selector=recipient_list like'%,aqapp2,%'
agent_reply_subscriber_name	Specifies the subscriber name used when multiple adapter instances are used for the given application and given partition. This parameter is optional if only one instance is running. Possible value: The application name (parameter:application) concatenated with the instance number (parameter:instance_number) Default value: None	If application=ftpapp and instance_number=2, then agent_reply_subscriber_name=ftpapp2
agent_subscriber_name	Specifies the subscriber name used when this adapter registers its subscription. Possible value: A valid Oracle Advanced Queue subscriber name Default value: None	agent_subscriber_name=ftpapp
agent_throughput_measurement_enabled	Specifies if the throughput measurement is enabled. Set this parameter to true to set the throughput measurements. Default value: true.	agent_throughput_measurement_enabled=true
agent_tracking_enabled	Specifies if message tracking is enabled. Set this parameter to false to turn off tracking of messages. Set this parameter to true to track messages with tracking fields set in iStudio. Default value: true	agent_tracking_enabled=true

Table 2–7 (Cont.) adapter.ini Parameters

Parameter	Description	Example
`agent_use_custom_hub_dtd`	Specifies whether to use a custom DTD for the common view message when handing it to the hub. By default, adapters use a specific OracleAS Integration InterConnect DTD for all messages sent to the hub. Set this parameter to `true` to have the adapter use the DTD imported for the message of the common view instead of the OracleAS Integration InterConnect DTD. Default value: None	`agent_use_custom_hub_dtd=false`
`application`	Specifies the name of the application to which this adapter connects. This must match the name specified in iStudio while creating metadata. Possible value: An alphanumeric string Default value: None	`application=ftpapp`
`encoding`	Specifies the character encoding for published messages. The adapter uses this parameter to generate encoding information for the encoding tag of transformed OracleAS Integration InterConnect messages. OracleAS Integration InterConnect represents messages internally as XML documents. Possible value: A valid character encoding Default value: `UTF-8` When there is no existing encoding in the subscribed message, this parameter will be used to explicitly specify the encoding of the published message. This parameter will be ignored when the encoding already exists in the subscribed message.	`encoding=Shift_JIS`
`external_dtd_base_url`	Specify the base URL for loading external enitites and DTDs. This specifies to the XML parser to resolve the external entities in the instance document using the given URL. Possible value: A URL Default value: The URL of the current user directory	`external_dtd_base_url=file://C:\ORACLEHOME\Integration\InterConnect\adapters\AQApp\`
`instance_number`	Specifies the instance number to which this adapter corresponds. Specify a value only if you have multiple adapter instances for the given application with the given partition. Possible value: An integer greater than or equal to `1` Default value: None	`instance_number=1`
`nls_country`	Specifies the ISO country code. The codes are defined by ISO-3166. Possible value: A valid code. A full list of the codes is available at `http://www.chemie.fu-berlin.de/diverse/doc/ISO_3166.html` Default value: `US` **Note**: This parameter specifies date format and is applicable only for the date format.	`nls_country=US`

Table 2–7 (Cont.) adapter.ini Parameters

Parameter	Description	Example

nls_date_format	Specifies the format for a date field expressed as a string.
	Possible value: A valid date format pattern as shown in Table 2–8 for the definitions of the format characters.
	Default value: EEE MMM dd HHmmss zzz yyyy
		Date format pattern dd/MMM/yyyy can represent 01/01/2003.
		nls_date_format=dd-MMM-yy
		Multiple date formats can be specified as num_nls_formats=2
		nls_date_format1=dd-MMM-yy
		nls_date_format2=dd/MMM/yy

nls_language	Specifies the ISO language code. The codes are defined by ISO-639. Possible value: A valid code. A full list of these codes is available at http://www.ics.uci.edu/pub/ietf/http/related/iso639.txt Default value: en **Note**: This parameter specifies date format and is applicable only for the date format.	nls_language=en
partition	Specifies the partition this adapter handles as specified in iStudio. Possible value: An alphanumeric string Default value: None	partition=germany
service_class	Specifies the entry class for the Windows service. Possible value: oracle/oai/agent/service/AgentService Default value: None	service_class=oracle/oai/agent/service/AgentService
service_classpath	Specifies the class path used by the adapter JVM. If a custom adapter is developed and the adapter is to pick up any additional jar files, then add the files to the existing set of jar files. Possible value: A valid PATH setting Default value: None This parameter is for only Microsoft Windows.	service_classpath=D:\oracle\oraic\integration\interconnect\lib\oai.jar;D:\oracle\oraic\jdbc\classes12.zip
service_jdk_dll	Specifies the Dynamic Link Library (DLL) that the adapter JVM should use. Possible value: A valid jvm.dll Default value: jvm.dll This parameter is only for Microsoft Windows.	service_jdk_dll=jvm.dll
service_jdk_version	Specifies the JDK version that the adapter JVM should use. Possible value: A valid JDK version number Default value: 1.4 This parameter is only for Microsoft Windows.	service_jdk_version=1.4

Table 2–7 (Cont.) adapter.ini Parameters

Parameter	Description	Example
`service_max_heap_size`	Specifies the maximum heap size for the adapter JVM. Possible value: A valid JVM heap size Default value: `536870912` This parameter is only for Microsoft Windows.	`service_max_heap_size=536870912`
`service_max_java_stack_size`	Specifies the maximum size to which the JVM stack can grow. Possible value: A valid JVM maximum stack size Default value: Default value for the JVM This parameter is only for Microsoft Windows.	`service_max_java_stack_size=409600`
`service_max_native_stack_size`	Specifies the maximum size to which the JVM native stack can grow. Possible value: A valid JVM maximum native stack size Default value: Default value for the JVM This parameter is only for Microsoft Windows.	`service_max_native_size=131072`
`service_min_heap_size`	Specifies the minimum heap size for the adapter JVM. Possible value: A valid JVM heap size Default value: `536870912` This parameter is only for Microsoft Windows.	`service_min_heap_size=536870912`
`service_num_vm_args`	Specifies the number of `service_vm_arg`*`number`* parameters specified in JVM. Possible value: The number of `service_vm_arg`*`number`* parameters Default value: None This parameter is only for Microsoft Windows.	`service_num_vm_args=1`
`service_path`	Specifies the environment variable `PATH`. The `PATH` variable is set before starting the Java Virtual Machine (JVM). Typically, list all directories that contain necessary DLLs. Possible value: The valid `PATH` environment variable setting Default value: None This parameter is only for Microsoft Windows.	`service_path=%JREHOME%\bin;D:\oracle\oraic\bin`

`service_vm_arg`*`number`*	Specifies any additional arguments to the JVM. For example, to retrieve line numbers in any stack traces, set `service_vm_arg1=java.compiler=NONE`. If a list of arguments exists, then use multiple parameters as shown in the example, by incrementing the last digit by `1`.
	Possible value: A valid JVM arguments
	Default value: None
	This parameter is only for Microsoft Windows.
		`service_vm_arg1=java.compiler=NONE`
		`service_vm_arg2=oai.adapter=.aq`

Table 2–8 shows the reserved characters used to specify the value of the `nls_date_format` parameter. Use these characters to define date formats.

Table 2–8 Reserved Characters for the nls_date_format Parameter

Letter	Description	Example
G	Era designator	`AD`
y	Year	`1996` or `96`
M	Month in year	`July` or `Jul` or `07`
w	Week in year	`27`
W	Week in month	`2`
D	Day in year	`189`
d	Day in month	`10`
F	Day of week in month	`Number 2`
E	Day in week	`Tuesday` or `Tue`
a	a.m./p.m. marker	`P.M.`
H	Hour in day (0-23)	`0`
k	Hour in day (1-24)	`24`
K	Hour in a.m./p.m. (0-11)	`0`
h	Hour in a.m./p.m. (1-12)	`12`
m	Minute in hour	`30`
s	Second in minute	`55`
S	Millisecond	`978`

FTP Adapter-Specific Parameters

Table 2–9 lists the parameters specific to the FTP adapter. With the exception of the `bridge_class` parameter, all parameters can be edited after installation.

Table 2–9 FTP Adapter-Specific Parameters

Parameter	Description	Example
`bridge_class`	Specifies the entry class for the FTP adapter. A value must be specified and cannot be modified later. Possible value: `oracle.oai.agent.adapter.technology.TechBridge` Default value: None	`bridge_class=oracle.oai.agent.adapter.technology.TechBridge`
`file.receiver.max_msgs_retrieved`	Defines the maximum number of messages to be retrieved in each session. Default value: `30`	`file.receiver.max_msgs_retrieved = 10`
`file.receiver.polling_interval`	Defines the time interval to poll the message source, in milliseconds. Default value: 60000	`file.receiver.polling_interval = 10000`
`file.receiver.proxy_host`	Specifies the name of the computer that servers as the proxy server for the inbound FTP server. Possible value: A correct host name Default value: None	`file.receiver.proxy_host=www-proxy.foo.com`

Table 2–9 (Cont.) FTP Adapter-Specific Parameters

Parameter	Description	Example

`file.receiver.read_only`	Specifies whether the inbound directory has read-only or read-write privileges. When the value of this parameter is `false`, the Inbound directory is read-write enabled, and the retrieved files are deleted from the inbound directory. However, when the value of this parameter is `true`, the Inbound directory is read-only, and the retrieved files will not be deleted from the inbound directory.
	Possible value: `true` or `false`
	Default value: `false`
		`file.receiver.read_only=true`
		`file.receiver.read_only=false`

`file.receiver.read_ordered`	Specifies whether files should be retrieved in the same order in which they were created. If the value of `file.receiver.read_only` parameter is `true`, then the `file.receiver.read_ordered` parameter is also implicitly `true`.
	Possible value: `true` or `false`
	Default value: `false`
		`file.receiver.read_ordered=true`
		`file.receiver.read_ordered=false`

`file.receiver.read_ordered.timestamp.source`	Specifies how the FTP adapter should retrieve the last-modified time from the files.
	Possible value: *`timestamp`*, *`listtime`*, or *`filename`*
	where
	■ `timestamp` refers to time returned by the FTP command `MDTM`. For file mode, `timestamp` refers to the Operating System time stamp.
	■ `listtime` refers to the time returned by FTP command `LIST`.
	■ `filename` refers to the time stamp embedded in the file name.
	Default value: `listtime`
		`file.receiver.read_ordered.timestamp.source=listtime`
		`file.receiver.read_ordered.timestamp.source=timestamp`
		`file.receiver.read_ordered.timestamp.source=filename`

Table 2–9 (Cont.) FTP Adapter-Specific Parameters

Parameter	Description	Example

`file.receiver.read_ordered.timestamp.format`	Specifies the string format of the dates returned from the specified source where source is the value specified in the `file.receiver.read_ordered.timestamp.source` parameter.
	Note: If there is more than one format, then all formats should be specified.
	Possible value:
	■ For `timestamp` source: (required only for FTP)
	`file.receiver.read_ordered.timestamp.format=` *<Starting position(column number)of the timestamp>, <ending position(column number)of the timestamp>, <timestamp format>*
	■ For `listtime` source
	`file.receiver.read_ordered.timestamp.format=` *<Starting position(column number)of the timestamp>, <ending position(column number)of the timestamp>, <timestamp format>*
	■ For `filename` source:
	file.receiver.read_ordered.timestamp.format=8,21,'N'
	`file.receiver.read_ordered.timestamp.format=` *<Starting position(column number)of the timestamp>, <ending position(column number)of the timestamp>, <timestamp format>*
	Default value: The Sun Solaris masks
		Sun Solaris FTP:
		`file.receiver.read_ordered.timestamp.format1=41,53,'MMM dd HH:mm',ucy`
		where `ucy` means Use Current Year. The time format used in this example does not display year. You can use `ucy` to refer to year of a date.
		`file.receiver.read_ordered.timestamp.format2=41,53,'MMM dd yyyy'`
		WAR-FTP:
		`file.receiver.read_ordered.timestamp.format1=29,41,'MMM dd HH:mm',ucy`
		`file.receiver.read_ordered.timestamp.format2=29,41,'MMM dd yyyy’`

`file.receiver.read_ordered.timestamp.offset`	Specifies the time offset to correct the time returned by the FTP server. Possible value: A valid time offset in milliseconds Default value: 0	`file.receiver.read_ordered.timestamp.offset=2653200000`
`file.receiver.read_ordered.polling_gap`	Specifies the duration in milliseconds for which a file should remain in the inbound directory before it is retrieved. This ensures that only completely written files are retrieved. Possible value: A valid duration in miliseconds Default value: `5000`	`file.receiver.read_ordered.polling_gap=0`

Table 2–9 (Cont.) FTP Adapter-Specific Parameters

Parameter	Description	Example
`file.receiver.user`	Specifies the FTP user name for the inbound FTP server. Possible value: A valid FTP user name Default value: None	`file.receiver.user=joe`
`file.sender.customizer_class`	Specifies the class name for customization used by the file sender. Default value: `oracle.oai.agentt.adapter.technology.FileDefaultSenderCustomizer`	`file.sender.customizer_class=MyFileSenderCustomizer`
`file.sender.file_name_rule`	Specifies the rule for generating file names used by the file sender. Default value: `%APP%%PART%_%TIME%`	`file.sender.file_name_rule=%APP%_%EVENT%_%TIME%.xml`
`file.sender.proxy_host`	Specifies the name of the computer that serves as the proxy server for the outbound FTP server. Possible value: A correct host name Default value: None	`file.sender.proxy_host=www-proxy.foo.com`
`file.sender.proxy_port`	Specifies the port number of the proxy server for the outbound FTP server. Possible value: A valid port number Default value: None	`file.sender.proxy_port=80`
`file.sender.staging_dir`	Specifies the staging directory name for the file sender. Default value: None	`file.sender.staging_dir=/private/ipdev1/ftpstaging1`
`file.sender.type`	Indicates the file types. Possible value: ASCII or binary Default value: binary	`file.sender.type=ASCII`
`file.sender.user`	Specifies the FTP user name for the outbound FTP server. Possible value: A valid FTP user name Default value: None	`file.sender.user=joe`
`ota.d3ls`	Specifies the list of data definition description language (D3L) XML files used by the bridge. Each business event handled by the bridge must have its own D3L XML file. Whenever a new D3L XML file is imported in iStudio for use by an application using the FTP adapter, the parameter must be updated and the FTP adapter restarted. Default value: None	`ota.d3ls=person.xml, person1.xml`

Table 2–9 (Cont.) FTP Adapter-Specific Parameters

Parameter	Description	Example

`ota.receive.endpoint`	Defines the FTP receiving endpoint URL. The URL is written as follows: `ftp://host name/directory path` or `file://localhost/directory path`.
	Possible value: `ftp://host name/directory path`.
	Default value: None
	On Windows platforms, if you have a file system endpoint, then you can use the drive letter and directory name as part of the endpoint URL.
		`ota.receive.endpoint=ftp://ip-sun/private/ipdev1/test/inbound`
		On Windows:
		`ota.receive.endpoint=file://localhost/c:\test`

`ota.send.endpoint`	Defines the FTP sending endpoint URL. The URL is written as follows: `ftp://host name/directory path` or `file://localhost/directory path`. Possible value: `ftp://host name/directory path` Default value: None **Note**: Do not set the `ota.send.endpoint` parameter to the same value as the `ota.receive.endpoint` parameter unless you perform a loop-back test for the FTP adapter. The files sent are used by the receiving end.	`ota.send.endpoint=ftp://ip-sun/private/ipdev1/test/inbound`
`ota.time_format`	Specifies the format in which you want to define the time stamp. If you don't want to define the `smtp_sender_customizer_class` parameter, but still want to format file name generation, then use this parameter. Possible value: The format options are identical to `java.text.SimpleDateFormat` Default value: None	`ota.time_format=yyyyMMddHHmmss`
`ota.time_format_couuntry_code`	Specifies the country code. If `ota.time_format` is null, then the other two will be ignored and the timestamp will be in milliseconds. If `ota.time_format` is not null and if either this parameter or the `ota.time_format_language_code` parameter is null, then the default value for both the parameters is used, ignoring the values given. Possible value: A valid ISO country code as mentioned in `java.util.Locale` Default value: None	`ota.time_format_couuntry_code=FR`

Table 2–9 (Cont.) FTP Adapter-Specific Parameters

Parameter	Description	Example
`ota.time_format_language_code`	Specifies the language code. If `ota.time_format` is null, then this parameter will be ignored and the timestamp will be in milliseconds. If `ota.time_format` is not null and if either this parameter or `ota.time_format_couuntry_code` parameter is null, then the default value for both the parameters is used, ignoring the values given. Possible value: A valid ISO language code as mentioned in `java.util.Locale` Default value: None	`ota.time_format_language_code=fr`
`ota.type`	Defines the type of payload this adapter handles. Possible values: `XML` and `D3L` Default value: None	`ota.type=XML`

`file.receiver.exception_dir`	Specifies a URL which represents either an FTP directory or a file location. An FTP URL can be specified for the exception directory only if the receiving endpoint is also an FTP URL and the host name in the URL is the same. When a processing exception occurs, the host name, user name, and password of the receiving endpoint will be used to log on to the FTP server to store the messages that are not processed successfully. Ensure that this directory exists on the FTP server (or the local file system if file URL is used) and is writable by the FTP adapter process.
	Possible value: A URL which represents either an FTP directory or a file location
	Default value: None
		`file.receiver. execption_dir=ftp://acme.com/private/user/error`
		or
		`file.receiver.exception_dir=file://localhost/private/user/error`

`file.receiver.proxy_port`	Specifies the port number of the proxy server for the inbound FTP server. Possible value: A valid port number Default value: None	`file.receiver.proxy_port=80`
`file.receiver.customizer_class`	Specifies the class name for customization used by the file receiver. Default value: `oracle.oai.aget.adapter.technology.DefaultReceiverCustomizer`	`file.receiver.customizer_class=MyFileReceiverCustomizer`

2.4 Uninstalling the FTP Adapter

To uninstall the FTP adapter, perform the following:

1. Navigate to the *`MiddleTier`*`\opmn\bin` directory.
2. Run the following command to check the adapter status.

   ```
   opmnctl status
   ```

3. If the FTP adapter instance that you want to remove is running, stop it by using the the following command:

```
opmnctl stopproc ias-component="InterConnect" process-type="myFTPAPP"
```

where myFTPAPP is the name of the adapter instance.

4. Navigate to the *MiddleTier*\bin directory and run the following command to stop the Enterprise Manager:

```
emctl stop iasconsole
```

5. Carefully, remove the adapter process-type entry from the opmn.xml file located in the *MiddleTier*\opmn\conf directory. For example, to remove an FTP adapter instance myFTPApp1, delete the following information specific to the adapter instance:

```
<process-type id="myFTPApp1" module-id="adapter" working-dir="$ORACLE_
HOME/integration/interconnect/adapters/myFTPApp1" status="enabled">
       <start timeout="600" retry="2"/>
       <stop timeout="120"/>
       <port id="icadapter_dmsport_range" range="15701-15800"/>
       <process-set id="myFTPApp1" restart-on-death="true" numprocs="1">
            <module-data>
                <category id="start-parameters">
                    <data id="java-parameters" value="-Xms8M"/>
                    <data id="class-name"
                     value="oracle.oai.agent.service.AgentService"/>
                </category>
                <category id="stop-parameters">
                    <data id="java-parameters" value="-mx64m"/>
                    <data id="class-name"
                     value="oracle.oai.agent.proxy.ShutdownAgent"/>
                    <data id="application-parameters"
                     value="persistence/Agent.ior"/>
                </category>
            </module-data>
       </process-set>
</process-type>
```

6. Save the opmn.xml file.

7. Navigate to the *MiddleTier*\opmn\bin directory and run the following command to reload the OPMN:

```
opmnctl reload
```

8. Navigate to the *ORACLE_HOME*\integration\interconnect\adapters directory and delete the folder that was created for the removed adapter instance.

9. Navigate to the *MiddleTier*\bin directory and run the following command to start the Enterprise Manager:

```
emctl start iasconsole
```

3

Design-Time and Run-Time Concepts

This chapter describes the design-time and run-time concepts for the FTP adapter. It contains the following topics:

- FTP Adapter Design-Time Concepts
- FTP Adapter Run-Time Concepts
- Customizing the FTP Adapter
- Starting the FTP Adapter
- Stopping the FTP Adapter

3.1 FTP Adapter Design-Time Concepts

The FTP adapter can handle XML and D3L structured payloads, such as pure XML data with strings beginning with `<xml...`, and binary data described by a D3L XML file.

3.1.1 XML Payload Type

You can import a Document Type Definition (DTD) or XML Schema Definition in iStudio, which determines how the FTP adapter parses a received XML document into an OracleAS Integration InterConnect application view event. In addition, you can use the DTD or XSD to describe how an inbound application view message is converted to an XML document. Use the message type option XML when defining a new integration point in any of the event wizards.

Ensure that the `ota.type` parameter in the `adapter.ini` file is set to `XML`, instead of `D3L`.

When the FTP adapter operates in the XML payload mode, no transformations are performed on the messages between native view and application view. Any Extensible Stylesheet Language Transformations (XSLT) should be performed either before sending an XML document to OracleAS Integration InterConnect or after receiving one from OracleAS Integration InterConnect.

3.1.2 D3L Payload

The FTP adapter performs a two-way conversion and transformation of messages between application view and native format.

An application based on the FTP adapter can use the iStudio Message Type D3L and the iStudio D3L Data Type Import options when importing a data type. In this case, messages received or sent by the FTP adapter must adhere to the fixed byte-level layout defined in a D3L XML file.

The D3L Data Type Import option can also define common view data types.

> **See Also:** *Oracle Application Server Integration InterConnect User's Guide*, Appendix B, for additional information on D3L

3.2 FTP Adapter Run-Time Concepts

This section describes the key run-time components of the FTP adapter. This section contains the following topics:

- FTP Receiver
- FTP Sender

3.2.1 FTP Receiver

The FTP adapter can receive messages from a single receiving endpoint, which is either a remote FTP server or a local file system. The receiving endpoint is in the following form:

- For an FTP protocol: `ftp://host name/directory path`
- For a local file system: `file://localhost/directory path`

The agent converts the application view event into a common view event and passes it to OracleAS Integration InterConnect for further routing and processing. Once the message is successfully passed to OracleAS Integration InterConnect, the corresponding FTP file on the remote FTP server or local file system is removed. An exception directory on the remote FTP server or local file system can be specified for storing the unsuccessfully processed files. If no exception directory is specified, then the file will be discarded. The archive of exception files of the FTP adapter is appended with a time stamp *`original filename_timestamp`*. This is to avoid the two incoming files having the same file name. The properties for the FTP receiver available in the `adapter.ini` in `file.receiver.*` format:

> **Note:** The value of the exception directory should be a URL which represents either an FTP directory or a file location. An FTP URL can be specified for the exception directory if the receiving endpoint is also an FTP URL. The host name in the URL is the same. When a processing exception occurs, the host name, user name, and password of the receiving endpoint will be used to log on to the FTP server to store the messages that are not processed successfully. Ensure that this directory exists on the FTP server (or the local file system if file URL is used) and is writable by the FTP adapter process.

> **Note:** The adapter subscribing to an event should be started before any other adapter can publish that event. If you publish an event before starting the subscribing adapter, then the event would not be delivered to the subscribing adapter.

3.2.2 FTP Sender

The FTP adapter comprises the FTP bridge and the run-time agent. When the agent has a message to send to an endpoint, the bridge is notified. The bridge then uses D3L or XML to perform the conversion of the common view object to the native format. The native format message is then sent through the FTP transport layer to an FTP endpoint. The FTP endpoint is written as follows:

```
ftp://host name/directory path
```

The file name at the destination site is automatically generated by the adapter and is in the following syntax:

```
ftp adapter name partition-time stamp
```

You can specify a rule for generating the file name when the FTP adapter sends a file to a directory or FTP server. To use this feature, add the parameter `file.sender.file_name_rule` to the `adapter.ini` file. The adapter recognizes the following tokens:

- `%APP%`: application name
- `%BO%`: business object name
- `%EVENT%`: corresponding event name
- `%MV%`: message version
- `%PART%`: partition number
- `%TIME%`: time stamp
- `%TYPE%`: message type

For example, `file.sender.file_name_rule=%APP%_%EVENT%_%TIME%.xml`rule instructs the adapter to generate files with the following pattern:

```
your app name_event name_current time stamp.xml
```

The FTP adapter supports sending outgoing messages from OracleAS Integration InterConnect to multiple FTP endpoints. The multiple endpoints feature enables sending messages to various FTP servers.

An endpoint can be associated with a subscribing event in iStudio by adding transport properties such as FTP endpoint, FTP user name, and endpoint password as metadata for the event. This is done using the Modify Fields function of the Subscribe Wizard - Define Application View dialog box. After associating an endpoint and an event, the message from the subscribing event is sent to the FTP endpoint.

When using the multiple-endpoint feature with `XML` data type, select the `Generic` event type instead of `XML`. The `Generic` event type enables you to enter the metadata for the endpoints using the Modify Fields feature associated with iStudio.

Table 3–1 shows how metadata is associated to an event called `SendOrder` that sends an order to an FTP server at foo.com in the `/private/user/test` directory.

Table 3–1 SendOrder Event Metadata

Parameter	Description
`file.sender.user=joe`	Defines the FTP user name
`ota.endpoint=sendOrderAppEP`	Specifies a unique endpoint name set in iStudio
`ota.send.endpoint=http://foo.com/servlet/test`	Specifies the sending endpoint for the FTP adapter

Note: The sender properties are not inherited from the adapter.ini file.

See Also: *Oracle Application Server Integration InterConnect User's Guide*

To avoid external applications picking up partial files, files that are not completely transferred, specify a staging directory. Set the file.sender.staging_dir parameter in the adapter.ini file. This parameter should contain only the directory name, for example, file.sender.staging_dir=/private/ipdev/test/staging.

Note: Do not use file or FTP URL, such as ftp://... or file://.... If the staging directory is used for an FTP server, then the path specified is a directory on the FTP server. Ensure that the path for the staging directory is valid.

The properties for the FTP receiver are available in the adapter.ini file in file.sender.* format.

3.3 Customizing the FTP Adapter

You can customize the adapter by implementing the following interfaces:

- oracle.oai.adapter.agent.technology.ReceiverCustomizer
- oracle.oai.adapter.agent.technology.FileSenderCustomizer

3.3.1 The ReceiverCustomizer Interface

You can use the ReceiverCustomizer interface to customize the TransportMessage object that the FTP adapter receives. The TransportMessage object represents the native message that the transport layer receives or sends.

- To customize the TransportMessage object, use the customizeTransportMessage() method. This method is called before the adapter processes the TransportMessage object.
- To modify the message, implement the customizeTransportMessage() method. You must also implement the createReplyMessage() method and ensure that it returns a null value.

The following code describes the file structure of the ReceiverCustomizer interface.

```
package oracle.oai.agent.adapter.technology;
import oracle.oai.agent.adapter.transport.TransportMessage;
import oracle.oai.agent.adapter.sdk.Agent;
public interface ReceiverCustomizer {

   public void customizeTransportMessage(Agent agent,
                                         int receiverType,
                                         TransportMessage transportMessage);

   public String createReplyMessage(Agent agent,
                                    int status,
                                    TransportMessage receivedTransportMessage);
}
```

The following table summarizes ReceiverCustomizer interface.

Methods	Description
createReplyMessage();	Creates a reply message based on the status of the message received. It contains the following parameters: agent: Log a message status: Status of the message process. If the value is TransportResponse.TRANSPORT_ACK, then the message is processed successfully. If the value is TransportResponse.TRANSPORT_ERROR, then the message has not been processed unsuccessfully. receivedTransportMessage: Transport message is received by the adapter. This parameter is used to transport headers in the transport message to create a meaningful message. The return string contains the reply message. This method is used for backward compatibility for the FTP adapter. However, for the FTP adapter, you should return a null value in this method.
customizeTransportMessage();	Enables you to customize the transport message received by the adapter. It uses the following parameters: agent: Log a message receiverType: Information on the type of adapter transportMessage: Customize the transport message received by the adapter

Example 3–1 Example of ReceiverCustomizer

The MyReceiverCustomizer class removes the first line in the native message.

```
import oracle.oai.agent.adapter.sdk.Agent;
import oracle.oai.agent.adapter.transport.TransportMessage;
import oracle.oai.agent.adapter.transport.TransportException;
import oracle.oai.agent.adapter.technology.ReceiverCustomizer;

public class MyReceiverCustomizer implements ReceiverCustomizer {
```

This example describes how to remove an extra line from the incoming files.

```
   public void customizeTransportMessage(Agent agent, int receiverType,
                                        TransportMessage transportMessage)
   {
      String payload = transportMessage.getBodyAsString();

      agent.logTraceMessage("payload received = " + payload, null, null, null);
//the following syntax removes the first line from the payload.
      String newPayload = removeFirstLine(payload);

      try {
         transportMessage.setBody(newPayload);
      }
      catch(TransportException te) {
      . . . .
      }
   }

   public String createReplyMessage(Agent agent, int status,
                                   TransportMessage receivedTransportMessage)
   {
```

```
   return null;
   }
}
```

List of Methods for the TransportMessage Class

The following table provides a list of methods you may choose for the TransportMessage class.

Method	Description
`public byte[] getBodyAsBytes();`	Get the body of the message as a byte array. Return the message in byte[].
`public InputStream getBodyAsInputStream();`	Get the body of the message and return an InputStream object representing the body of the message.
`public Properties getTransportHeaders();`	Get all transport-specific headers and return a Properties object that contains all the transport headers.
`public String getBodyAsString();`	Get the body of the message as String object. Return the message in String object.
`public String toString();`	Dump messages and headers.
`public void setBody(InputStream in) throws TransportException;`	Set the body of the message. The body type will be set to bytes. Parameters: InputStream: Contains the message. It throws an exception of TransportException type.
`public void setBody(String body) throws TransportException;`	Set the body of the message. The body type will be set to String. Parameters: body: body of the message It throws an exception of TransportException type

3.3.2 The FileSenderCustomizer Interface

You can use the FileSenderCustomizer interface to customize the file name and payload of the TransportMessage object that is sent to the transport layer.

The FileSenderCustomizer interface extends the SenderCustomizer interface. You must implement the FileSenderCustomizer interface by implementing the following two methods:

- FileSenderCustomizer.customizeTransportMessage()
- FileSenderCustomizer.generateFileName()

If you do not want to implement the generateFileName() method, then you can create a class that extends the oracle.oai.agent.adapter.technology.FileDefaultSenderCustomizer class, which is provided in the oai.jar file. In this case, you need to implement only the customizeTransportMessage() method.

3.3.2.1 The SenderCustomizer Interface

The following code describes the file structure of the SenderCustomizer interface.

```
package oracle.oai.agent.adapter.technology;
```

```
import oracle.oai.agent.adapter.sdk.MessageObject;
import oracle.oai.agent.adapter.sdk.AttributeObject;
import java.util.Properties;
import oracle.oai.agent.adapter.sdk.Agent;
import oracle.oai.agent.adapter.transport.TransportMessage;

public interface SenderCustomizer {
   public void customizeTransportMessage(Agent agent,
                                         TransportMessage transportMessage,
                                         MessageObject mobj,
                                         AttributeObject aobj);
}
```

The customizeTransportMessage method

This method specifies how to customize the transport message for transporting the sender. The adapter creates a `TransportMessage` object for the transport layer to send, based on the `MessageObject` object sent by OracleAS Integration InterConnect.

This method contains the following parameters:

- `agent`: Log messages
- `transportMessage`: The `TransportMessage` object that the adapter has created for sending
- `mobj`: The `MessageObject` from OracleAS Integration InterConnect
- `aobj`: The `AttributeObject` from OracleAS Integration InterConnect

  This method does not return anything. You can change the payload with the transportMessage parameter.

This method does not return anything. You can change the payload with the `transportMessage` parameter.

3.3.2.2 The FileSenderCustomizer Interface

The following code describes the file structure of the `FileSenderCustomizer` interface.

```
package oracle.oai.agent.adapter.technology;
import java.util.Date;
import oracle.oai.agent.adapter.sdk.MessageObject;
import oracle.oai.agent.adapter.sdk.AttributeObject;
import oracle.oai.agent.adapter.sdk.Agent;

public interface FileSenderCustomizer extends SenderCustomizer {
   public String generateFileName (Agent agent,
                                   String rule,
                                   String app,
                                   String partitition,
                                   Date time,
                                   MessageObject mobj,
                                   AttributeObject aobj);
}
```

The generateFileName method

This method generates a file name for sending a file. It contains the following parameters:

- agent: Use the Agent object to log message
- rule: Rule for generating subject. This parameter is read from file.sender.file_name_rule in the adapter.ini file
- app: The application name
- partition: Partition
- time: The time the OracleAS Integration InterConnect object is received
- mobj: A MessageObject passed from OracleAS Integration InterConnect
- aobj: An AttributeObject passed from OracleAS Integration InterConnect

  This method returns a string representing the file name

3.4 Starting the FTP Adapter

The process for starting the adapter varies based on the operating system.

- To start the FTP adapter on UNIX:
  1. Change to the directory containing the start script.

     ```
     cd ORACLE_HOME/integration/interconnect/adapters/Application
     ```

  2. Type **start** and press **Enter**.
- To start the FTP adapter from Services on Windows:
  1. Access the Services window from the Start menu.
  2. Select the *OracleHomeOracleASInterConnectAdapter-Application* service.
  3. Start the service based on the operating system.

> **Note:** You can also start and stop the FTP adapter using the IC Manager. Refer to *Oracle Application Server Integration InterConnect User's Guide* for more details.

3.4.1 Log File of FTP Adapter

You can verify the start up status of the FTP adapter by viewing the log.xml files. The files are located in the time-stamped subdirectory of the log directory of the FTP adapter. Subdirectory names have the following form:

```
timestamp_in_milliseconds
```

The following is an example of the information about an FTP adapter that started successfully:

```
The Adapter service is starting..
Registering your application (FTPAPP)..
Initializing the Bridge oracle.oai.agent.adapter.technology.TechBridge..
Starting the Bridge oracle.oai.agent.adapter.technology.TechBridge..
Service started successfully.
```

3.5 Stopping the FTP Adapter

The process for stopping the adapter varies based on the operating system.

- To stop the FTP adapter on UNIX:

  1. Change to the directory containing the stop script.

     ```
     cd ORACLE_HOME/integration/interconnect/adapters/Application
     ```

  2. Type **stop** and press **Enter**.

- To stop the FTP adapter from Services on Windows:

  1. Access the Services window from the Start menu.
  2. Select the *`OracleHomeOracleASInterConnectAdapter-Application`* service.
  3. Stop the service based on the operating system.

  You can verify the stop status of the FTP adapter by viewing the `log.xml` files. These files are located in the time-stamped subdirectory of the `log` directory of the FTP adapter.

A

Frequently Asked Questions

This appendix provides answers to frequently asked questions about the FTP adapter.

- FTP Adapter Does Not Publish or Subscribe to Messages
- How do I know the FTP adapter has started properly?
- The FTP adapter did not start properly. What went wrong?
- The FTP adapter is not starting. What could be the reason?
- Is it possible to edit the FTP adapter configuration settings created during installation?
- When I change an element in iStudio, such as mappings, it seems like the FTP adapter is using old information. What is happening?
- How do I secure my passwords?
- How can I deliver a message to a specific partition of the publishing FTP adapter?

A.1 FTP Adapter Does Not Publish or Subscribe to Messages

The FTP adapter is not publishing or subscribing to the messages.

Problem 1

The Transport properties of the FTP adapter might not be correct.

Solution 1

Specify 2 as the value of the log level parameter `agent_log_level` in the `adapter.ini` file of the FTP adapter. Restart the adapter and try to publish a message or subscribe to a message. You can see a log starting with `TransportProperties.TransportProperties()`. This log contains all the transport properties for the FTP adapter in the form of name value pairs. Ensure that the values of these properties are correct.

Problem 2

You might not have created a publish or subscribe event in iStudio for the FTP adapter that corresponds to the message structure. The FTP adapter is not able to match the message data to any of the events defined during design time and therefore cannot publish or subscribe the message.

Solution 2

Create a publish or subscribe event for the FTP adapter in iStudio corresponding to the message structure.

Problem 3

The FTP server is not based on RFC standards.

Solution 3

Ensure that the FTP server is based on RFC standards.

A.2 How do I know the FTP adapter has started properly?

View the log.xml file located in the timestamped subdirectory of the FTP adapter log directory:

Platform	Directory
UNIX	*ORACLE_HOME*/integration/interconnect/adapters/*Application*/log/*timestamp_in_milliseconds*
Windows	*ORACLE_HOME*\integration\interconnect\adapters\Application\log\timestamp_in_milliseconds

If there are no exceptions, then the FTP adapter has started properly.

A.3 The FTP adapter did not start properly. What went wrong?

View the exceptions in the FTP adapter log file (log.xml). The exceptions should provide information about what went wrong. It is possible that the FTP adapter is unable to connect to the repository. Ensure the repository is started properly. The FTP adapter will connect to the repository once it is started properly. You do not need to restart the Adapter.

> **See Also:** *Oracle Application Server Integration InterConnect User's Guide* for instructions on starting the repository on UNIX and Windows

A.4 The FTP adapter is not starting. What could be the reason?

If you are starting the adapter for the first time or if you have set the agent_metadata_cache parameter to none or demand, then check if the repository is up and running. In addition, check if the hub database is up and running.

One reason can be that Oracle Wallet does not contain the password information corresponding to your application name. For example, during installation you defined the application name as myFTPApp. Later, you changed the application name in iStudio to FTPApp. In such case, you need to specify the password corresponding to the new application name FTPApp in the Oracle Wallet. You can create password by using the oraclewallet command.

> **See Also:** Section A.7, "How do I secure my passwords?"

A.5 Is it possible to edit the FTP adapter configuration settings created during installation?

Yes, edit the parameters in the adapter.ini file in the following directory:

Platform	Directory
UNIX	*ORACLE_HOME*/integration/interconnect/adapters/*Application*/
Windows	*ORACLE_HOME*\integration\interconnect\adapters\Application\

See Also: Chapter 2, "Installation and Configuration"

A.6 When I change an element in iStudio, such as mappings, it seems like the FTP adapter is using old information. What is happening?

The FTP adapter caches information from iStudio. The information is stored in the repository locally. If you change something in iStudio and want to view the change in the runtime, then you need to stop the FTP adapter, delete the FTP adapter cache files, and restart the FTP adapter.

The FTP adapter has a persistence directory which is located in the FTP adapter directory. Deleting this directory when the FTP adapter has been stopped should make it obtain the new metadata from the repository when started.

A.7 How do I secure my passwords?

OracleAS Integration InterConnect uses Oracle Wallet Manager to maintain system passwords. When you install OracleAS Integration InterConnect, Oracle Wallet Manager is also installed and a password store is created. All passwords used by OracleAS Integration InterConnect components are stored in the password store. The password is stored in the Oracle Wallet in the following format:

```
ApplicationName/password
```

The `ApplicationName` is the name of the application, which is extracted from the `adapter.ini` file of the corresponding adapter. In the `adapter.ini` file, the `application` parameter specifies the `ApplicationName` to which this adapter connects. The password for the application is also retrieved from the `adapter.ini` file.

The number of entries is dependent on the type of adapter. For example, Database adapter needs two entries whereas AQ Adapter needs only one entry. The following table lists the entries that will be created for each adapter:

Adapter	Entry In Oracle Wallet
AQ	*ApplicationName*/aq_bridge_password
HTTP	*ApplicationName*/http.sender.password
HTTP	*ApplicationName*/sender.wallet_password
SMTP	*ApplicationName*/smtp.receiver.password
MQ	*ApplicationName*/mq.default.password
FTP	*ApplicationName*/file.sender.password
FTP	*ApplicationName*/file.receiver.password
DB	*ApplicationName*/db_bridge_schema1_password

Adapter	Entry In Oracle Wallet
DB	*ApplicationName*/db_bridge_schema1_writer_ password

You can create, update, and delete passwords using the `oraclewallet` command. When you run the command, it prompts you for the admin password.

You can use the following commands to manage your passwords:

- List all passwords in the store

  ```
  oraclewallet -listsecrets
  ```

- Create a password

  ```
  oraclewallet -createsecret passwordname
  ```

  For example, to create a password for the hub schema:

  ```
  oraclewallet -createsecret hub_password
  ```

- View a password

  ```
  oraclewallet -viewsecret passwordname
  ```

  For example, to view the password for the hub schema:

  ```
  oraclewallet -viewsecret hub_password
  ```

- Update a password

  ```
  oraclewallet -updatesecret passwordname
  ```

  For example, to update the password for the hub schema:

  ```
  oraclewallet -updatesecret hub_password
  ```

- Delete a password

  ```
  oraclewallet -deletesecret passwordname
  ```

  For example, to delete the password for the hub schema:

  ```
  oraclewallet -deletesecret hub_password
  ```

A.8 How can I deliver a message to a specific partition of the publishing FTP adapter?

Scenario: FTP adapter has two partitions `PAR1` and `PAR2`. You want to deliver event `create_customer` to partition `PAR1` and `add_customer` event to partition `PAR2`.

Perform the following tasks:

1. Assign event `create_customer` to partition `PAR1` and event `add_customer` to partition `PAR2`.
2. Create two directories. For example, `DIR1` and `DIR2`.
3. Configure the adapter with partition name `PAR1` to poll `DIR1` and adapter with partition name `PAR2` to poll `DIR2`.

4. Publish the message for event `create_customer` to `DIR1` and the event `add_customer` to `DIR2`.

B

Example of the adapter.ini File

This appendix shows a sample `adapter.ini` file for the FTP adapter.

> **See Also:** Configuring the FTP Adapter on page 2-9 for additional information on `adapter.ini` configuration parameters

The following code is an example of the FTP `adapter.ini` file.

```
#include <../../hub/hub.ini>

// *************
// ** Adapter **
// *************

// Application (as created in iStudio) corresponding to this Adapter.
application=myFtpApp

// Partition (as created in iStudio) corresponding to this Adapter.
partition=

// If you have multiple adapter instances for a given application with the
// given partition, each Adapter should have an instance number.

//instance_number=2

// Bridge class
bridge_class=oracle.oai.agent.adapter.technology.TechBridge

ota.type=D3L


// define the ftp sending endpoint
// For ftp, ota.send.endpoint=ftp://host name/path name
// For file, ota.send.endpoint=file://host name/path name
//
ota.send.endpoint= ftp://foo.s.com/private/ipdev1/test/d3l/inbound

// define the ftp receiving endpoint
// For ftp, ota.send.endpoint=ftp://host name/path name
// For file, ota.send.endpoint=file://host name/path name
//
ota.receive.endpoint=ftp://foo.s.com/private/ipdev1/test/d3l/inbound


//-------------------------------------
// ftp Sender initialization variables
```

```
//---------------------------------------

// ftp user (mandatory if ftp is used)
// file.sender.user=ipdev1
file.sender.user=ipdev1


// ftp user password (mandatory if ftp is used)
//file.sender.password=ipdev1
file.sender.password=ipwelcome


// file type (ASCII or BINARY)
//file.sender.type=BINARY
file.sender.type=ASCII

// proxy host
//file.sender.proxy_host=

// proxy port
//file.sender.proxy_port=
//staging directory
//file.sender.staging_directory =/tmp

//sender customizer class
//file.sender.customizer_class = MySenderCustomizer

//---------------------------------------
// ftp receiver initialization variables
//---------------------------------------

// ftp user (mandatory if ftp is used)
//file.receiver.user=ipdev1
file.receiver.user=ipdev1

// ftp user password (mandatory if ftp is used)
//file.receiver.password=ipdev1
file.receiver.password=ipwelcome

// file type (ASCII or BINARY)
//file.receiver.type=BINARY
file.receiver.type=BINARY

// proxy host
//file.receiver.proxy_host=

// proxy port
//file.receiver.proxy_port=
//receiver customizer class
//file.receiver.customizer_class = MyReceiverCustomizer

// define where to put the file that cannot be processed properly.
//file.receiver.exception_dir=


// define how often to poll
// the message source (in milli seconds)
file.receiver.polling_interval=60000

// define maximum number of messages
// retrieved in each polling session
file.receiver.max_msgs_retrieved=30

// D3L initialization variables
ota.d3ls=person2.xml:person1.xml


// *************
// ** Agent  ***
// *************

// Log level (0 = errors only, 1 = status and errors, 2 = trace, status and
// errors).
agent_log_level=2

// Hub message selection information
agent_subscriber_name=myFTPApp
agent_message_selector=recipient_list like ’%,myFTPApp,%’
// Only provide values for the next two parameters if you have multiple Adapter
// instances for the given application with the given partition.

//agent_reply_subscriber_name=
//agent_reply_message_selector=

// Set this to false if you want to turn off all tracking of messages (if true,
// messages which have tracking fields set in iStudio will be tracked)

agent_tracking_enabled=true

// Set this to false if you want to turn off all throughput measurements
agent_throughput_measurement_enabled=true

// By default, Adapters use an OAI specific DTD for all messages sent to the Hub
//as other OAI Adapters will be picking up the messages from the Hub and know
// how to interpret them. This should be set to true if for every message, you
//would like to use the DTD imported for that message’s Common View instead
//of the OAI DTD. This should only be set to true if an OAI Adapter
//is *NOT* receiving the messages from the Hub.

agent_use_custom_hub_dtd=false

// Sets the metadata caching algorithm. The possible choices are startup (cache
everything at startup: this may take a while if there is a lot of metadata in
// your Repository), demand (cache metadata as it is used) or none (no caching: //
this will slow down performance.)
agent_metadata_caching=demand

// Sets the DVM table caching algorithm. The possible choices are startup (cache
// all DVM tables at startup: this may take a while if there are a lot of tables
// in your Repository), demand (cache tables as they are used) or none (no caching
//: this will slow down performance.)
agent_dvm_table_caching=demand

// Sets the lookup table caching algorithm. The possible choices are startup
// (cache all lookup tables at startup: this may take a while if there are a lot
// of tables in your Repository), demand (cache tables as they are used) or none
// (no caching: this will slow down performance.)
agent_lookup_table_caching=demand

// If metadata caching, DVM table caching, or lookup table caching are turned on
```

```
//(startup or demand) then the Adapter caches metadata or DVM tables it retrieves
//from the Repository in a file cache. When you restart the Adapter,it will not
// have to get that metadata or DVM table from the Repository again because it is
// in the cache files.However, if you change some metadata or DVM table using
// iStudio and you want the Adapter to use those changes the next time it is
// started you can either delete the cache files or set this parameter to true
// before restarting.
agent_delete_file_cache_at_startup=false

// Max number of application datatype information to cache
agent_max_ao_cache_size=200

// Max number of common datatype information to cache
agent_max_co_cache_size=100

// Max number of message metadata to cache
agent_max_message_metadata_cache_size=200

// Max number of DVM tables to cache
agent_max_dvm_table_cache_size=200

// Max number of lookup tables to cache
agent_max_lookup_table_cache_size=200

// Internal Agent queue sizes
agent_max_queue_size=1000
agent_Persistence_queue_size=1000

// Persistence
agent_persistence_cleanup_interval=60000
agent_persistence_retry_interval=60000


 //////////////////
 // End Comments //
//////////////////
```

Index