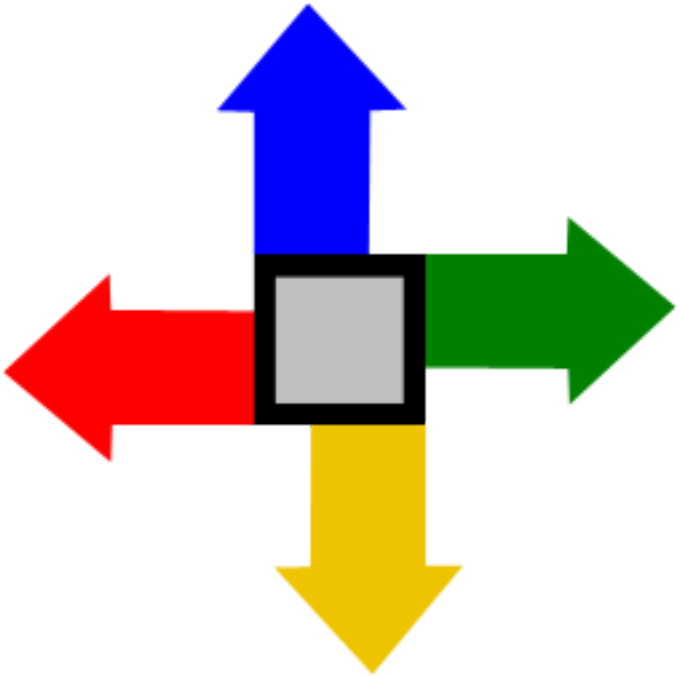
Universal File Mover Installation and Operation Manual



Capitalware Inc.
Unit 11, 1673 Richmond Street, PMB524
London, Ontario, Canada
N6G 2N3
sales@capitalware.com
http://www.capitalware.com

Table of Contents

1 Introduction

1.1 Overview

Universal File Mover (UFM) is more than a simple tool to manage the transfer of files. It allows the user to combine business processes into a workflow. A workflow describes the tasks, procedural steps, people involved, required input and output information, and tools required for each step in the business process. The user creates a UFM Workflow XML, which details the sequence of concatenated steps in a business process, by combining a series of Action commands. These Action commands define which actions are to be taken, the order of the actions, and how errors are to be handled. UFM processes the Action commands as per the UFM Workflow XML file.

UFM currently contains 42 Action commands. These action commands fall into 5 categories: IBM MQ Actions, Network Actions, File Actions, Control Actions and Other Actions.

- ***IBM MQ Actions***: MQSend, MQReceive and MQPutQuit
- ***Network Actions***: Ftp, HttpGetFile, HttpPutFile, Scp, SendEmail and SFtp
- ***File Actions***: Append, Convert, Copy, Delete, DecryptFile, EncryptFile, MakeDir, Merge, MergeSort, Move, ReEncode, RemoveDir, RemoveItems, Rename, ReplaceText, Sort, SortUnique, Split, Tar, Touch, Unique, UnTar, UnZip, Watch, WriteText and Zip
- ***Control Actions***: If/Else, Loop and Sleep
- ***Other Actions***: Echo, Execute, Launch and Schedule

UFM can transfer files in 1 of 5 ways: IBM MQ, FTP, SFTP, SCP and HTTP. For example, UFM can retrieve a file from a remote server via FTP and then send it via IBM MQ to another server. UFM can move/transfer files in any combination the user wishes.

UFM's MQSend Action sends a file via MQ and has capability to encrypt the data using Advanced Encryption Standard (AES) and also the ability to compress the data before it is sent. The MQReceive Action can decrypt and decompress the incoming data before it is written to the target file. UFM's EncryptFile Action uses AES to encrypt a file and the DecryptFile Action uses AES to decrypt the file.

Note: AES is a data encryption scheme, adopted by the US government, that uses three different key sizes (128-bit, 192-bit, and 256-bit). AES was announced by National Institute of Standards and Technology (NIST) as U.S. FIPS PUB 197 (FIPS 197) on November 26, 2001.

The user can define a centralized Status queue in the UFM Workflow XML, so that a status message is generated with the details of each Action as they are processed. Another program, UFM Status Monitor, can be used to view the Status queue for information on the outcome of each processed UFM Workflow.

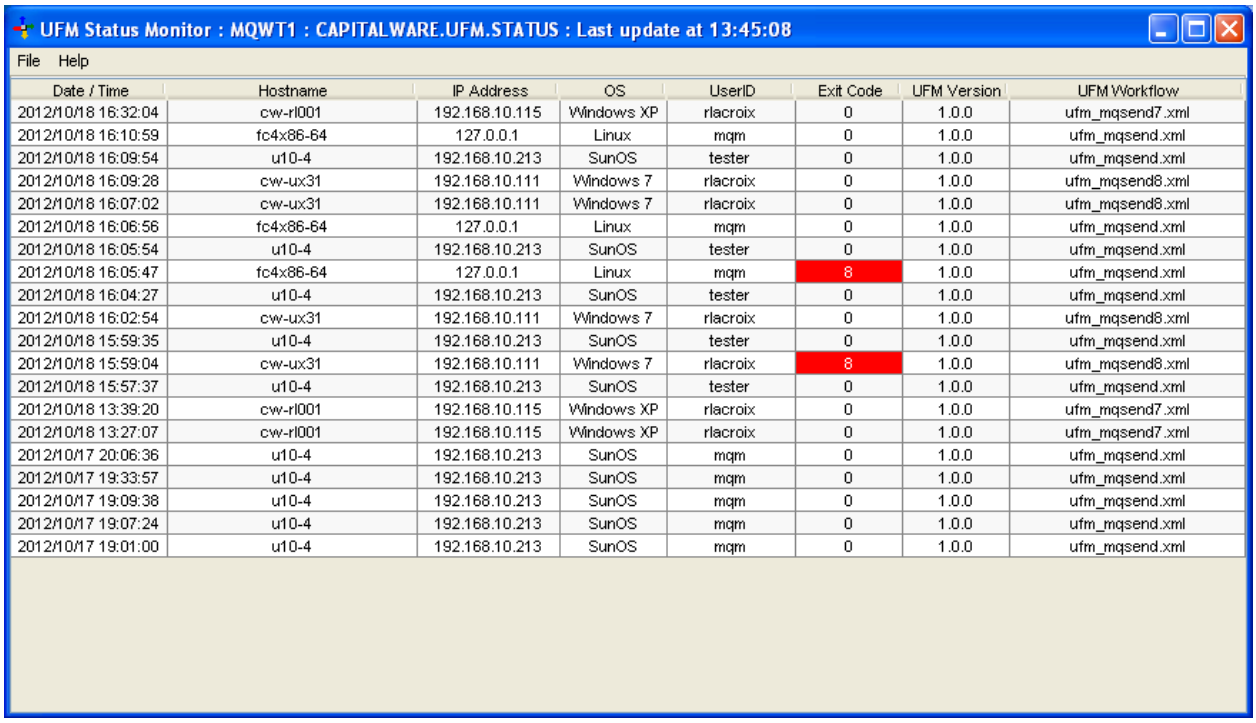
1.2 Executive Summary

The major features of UFM are as follows:

- A comprehensive set of Action commands used to create a workflow for processing files
- A single MQSend Action can send more than 1 file
- A single MQReceive action can receive more than 1 file
- Assured file delivery using IBM MQ
- MQSend and MQReceive Actions use (AES) to encrypt and decrypt the message data
- MQSend and MQReceive Actions support on-the-fly compression and decompression of the message data
- If Action is used to perform a conditional test against an action's variable
- Watch, MQReceive and Schedule Actions can be run as a Unix/Linux daemon or as a Windows service
- Has file based actions (i.e. Append, Copy, etc...)
- Launch and Schedule Actions can execute other UFM Workflow XML files
- Ability to launch external programs (Execute)
- Support for a centralized Status queue.
- Supports both styles of MQ security
- Provides complete logging capability
- Licensed under Apache License 2
- Free to use (support is extra)

1.3 Universal File Mover Status Monitor

Universal File Mover Status Monitor application is designed to display the UFM Workflow status messages from the UFM status queue. Please see the ***UFM-SM Installation and Operation*** manual for more details.



Date / Time	Hostname	IP Address	OS	UserID	Exit Code	UFM Version	UFM Workflow
2012/10/18 16:32:04	cw-rl001	192.168.10.115	Windows XP	rlacroix	0	1.0.0	ufm_mqsend7.xml
2012/10/18 16:10:59	fc4x86-64	127.0.0.1	Linux	mqm	0	1.0.0	ufm_mqsend.xml
2012/10/18 16:09:54	u10-4	192.168.10.213	SunOS	tester	0	1.0.0	ufm_mqsend.xml
2012/10/18 16:09:28	cw-ux31	192.168.10.111	Windows 7	rlacroix	0	1.0.0	ufm_mqsend8.xml
2012/10/18 16:07:02	cw-ux31	192.168.10.111	Windows 7	rlacroix	0	1.0.0	ufm_mqsend8.xml
2012/10/18 16:06:56	fc4x86-64	127.0.0.1	Linux	mqm	0	1.0.0	ufm_mqsend.xml
2012/10/18 16:05:54	u10-4	192.168.10.213	SunOS	tester	0	1.0.0	ufm_mqsend.xml
2012/10/18 16:05:47	fc4x86-64	127.0.0.1	Linux	mqm	8	1.0.0	ufm_mqsend.xml
2012/10/18 16:04:27	u10-4	192.168.10.213	SunOS	tester	0	1.0.0	ufm_mqsend.xml
2012/10/18 16:02:54	cw-ux31	192.168.10.111	Windows 7	rlacroix	0	1.0.0	ufm_mqsend8.xml
2012/10/18 15:59:35	u10-4	192.168.10.213	SunOS	tester	0	1.0.0	ufm_mqsend.xml
2012/10/18 15:59:04	cw-ux31	192.168.10.111	Windows 7	rlacroix	8	1.0.0	ufm_mqsend8.xml
2012/10/18 15:57:37	u10-4	192.168.10.213	SunOS	tester	0	1.0.0	ufm_mqsend.xml
2012/10/18 13:39:20	cw-rl001	192.168.10.115	Windows XP	rlacroix	0	1.0.0	ufm_mqsend7.xml
2012/10/18 13:27:07	cw-rl001	192.168.10.115	Windows XP	rlacroix	0	1.0.0	ufm_mqsend7.xml
2012/10/17 20:06:36	u10-4	192.168.10.213	SunOS	mqm	0	1.0.0	ufm_mqsend.xml
2012/10/17 19:33:57	u10-4	192.168.10.213	SunOS	mqm	0	1.0.0	ufm_mqsend.xml
2012/10/17 19:09:38	u10-4	192.168.10.213	SunOS	mqm	0	1.0.0	ufm_mqsend.xml
2012/10/17 19:07:24	u10-4	192.168.10.213	SunOS	mqm	0	1.0.0	ufm_mqsend.xml
2012/10/17 19:01:00	u10-4	192.168.10.213	SunOS	mqm	0	1.0.0	ufm_mqsend.xml

1.4 Action Commands Summary:

The UFM Workflow XML Action framework contains 40 Action commands.

IBM MQ Actions:

- **MQSend** - Sends 1 or more files as individual messages via MQ. Each file can be encrypted and/or compressed before it is sent. Each file can be sent to a single queue or multiple queues.
- **MQReceive** - Receives an incoming messages and writes them to a file (can be run as a daemon or as a Windows service). Receive Action will decrypt and/or decompress and encrypted and/or compressed messages.
- **MQPutQuit** - Puts a 'Quit' message on a queue (to stop the MQReceive Action running as a daemon)

Network Actions:

- **Ftp** – Get and/or put files using FTP network protocol
- **HttpGetFile** - Get a file using HTTP application protocol from a web server
- **HttpPutFile** - Put a file to a web server using HTTP application protocol
- **Scp** – Securely copy files to and/or from a remote host
- **SendEmail** – Sends an email to 1 or more recipients.
- **SFtp** - Get and/or put files using Secure FTP network protocol

File Actions:

- **Append** - Appends a file to another file
- **Convert** - Converts file format (i.e. CRLF <=> LF <=> CR and/or ASCII to EBCDIC)
- **Copy** - Copies 1 or more files from a directory to another directory
- **Delete** - Deletes 1 or more files in a directory
- **DecryptFile** - Decrypt a file using AES 128, 192 or 256-bit decryption
- **EncryptFile** - Encrypt a file using AES 128, 192 or 256-bit encryption
- **MakeDir** - Create a directory
- **Merge** - Merge 2 or more files to another file
- **MergeSort** - Merge 2 or more files and sort the data to another file
- **Move** - Moves 1 or more files from a directory to another directory
- **ReEncode** – Change the file's character encoding to another encoding
- **RemoveDir** - Remove a directory
- **RemoveItems** – Removes items in 1 list from another list
- **Rename** - Renames a file
- **ReplaceText** - Performs a search and replace of text in a file.
- **Sort** - Sorts the data of a file into another file
- **SortUnique** - Sorts the data of a file into another file then removes duplicate lines from the file.
- **Split** - Split a file into several files
- **Tar** - Combine a file(s) or a directory of files into a tar archive
- **Touch** - Update a file's the modification time or create the file if it does not exist
- **Unique** - Removes duplicate lines from a file.
- **UnTar** - Extract tar archive to a directory
- **UnZip** - Uncompress a zip archive to a directory

- **Watch** - Monitor for a particular file or monitor a directory for files to appear and then executes a series of Actions.
- **WriteText** – Writes the text to a file.
- **Zip** - Compresses a file(s) or a directory of files into a Zip archive

Control Actions:

- **If/Else** - Performs a conditional test against an action's variable
- **Loop** - Iterates over a group of Actions.
- **Sleep** – Pause the UFM Workflow for a period of time.

Other Actions:

- **Echo** – Echo a text string to log file.
- **Execute** - Runs an external program / application
- **Launch** - Invokes an UFM Workflow XML file.
- **Schedule** - Invokes an UFM Workflow XML file at a specific date and/or time.

UFM can connect to a IBM MQ queue manager in 3 possible ways:

- Locally in binding mode
- Remotely using a Client Channel Definition Table (CCDT)
- Remotely using a MQ XML file

UFM supports both forms of IBM MQ security:

- SSL for connecting to remote queue managers.
- 3rd party security exit for connecting to remote queue managers.

1.5 Prerequisites

This section provides the minimum supported software levels.

1.5.1 Java

Universal File Mover requires Java v7 or higher.

1.5.2 IBM MQ

Universal File Mover requires IBM MQ v7.0, v7.1, v7.5, v8.0, v9.0, v9.1 or higher.

1.6 File Definitions

The files used in UFM are defined as follows:

- ***UFM_Workflow*** XML file is the file that is defined by a ***UFM_Workflow.dtd***. It contains the Action commands
- ***UFM_Email_SMTP*** XML file is the file that is defined by a ***UFM_Email_SMTP.dtd***. It contains information to send an email message via SMTP.
- ***UFM_Ftp*** XML file is the file that is defined by a ***UFM_Ftp.dtd***. It contains information for getting and/or putting files to/from an FTP server.
- ***UFM_Holiday*** XML file is the file that is defined by a ***UFM_Holiday.dtd***. It contains exclude dates (holidays) where a schedule is to be skipped.
- ***UFM_Job*** XML file is the file that is defined by a ***UFM_Job.dtd***. It contains information to run an external program
- ***UFM_MQ*** XML file is the file that is defined by a ***UFM_MQ.dtd***. It contains the information to connect to a particular queue manager
- ***UFM_Schedule*** XML file is the file that is defined by a ***UFM_Schedule.dtd***. It contains the information for starting an UFM Workflow XML at a particular date and/or time.
- ***UFM_Scp*** XML file is the file that is defined by a ***UFM_Scp.dtd***. It contains information for sercurely copying files to/from a remote server.
- ***UFM_SFtp*** XML file is the file that is defined by a ***UFM_SFtp.dtd***. It contains information for sercurely getting and/or putting files to/from an SFTP server.
- ***service.properties*** file (*Windows only*) is used to relate a UFM Windows service name to a UFM Workflow XML file

2 Installing Universal File Mover

This section describes how to install Capitalware's Universal File Mover.

2.1 Windows Installation

To install Universal File Mover on Windows, do the following instructions:

- Run the install program called:
  - **ufm-setup-32bit.exe** for 32-bit JVM
  - **ufm-setup-64bit.exe** for 64-bit JVM
- The installer follows the standard Windows install procedures and provides default values for the user.
- When the install program has completed execution, there will be a newly created folder under ***Start*** -> ***All Programs*** called ***Universal File Mover***. This will open a Windows Command prompt.

Note: If the user did not install UFM in the default directory (i.e. C:\Capitaware\UFM\ ) then the user will need to edit the ***ufm.bat*** file and update the UFM_HOME batch variable.

The ***ufm.bat*** file expects the ***MQ_JAVA_LIB_PATH*** environment variable to be set via the install of either MQ Server or MQ Client software. If your system does not have the ***MQ_JAVA_LIB_PATH*** environment variable set then either set the environment variable to the correct location or edit line number 24 of the ***ufm.bat*** file to set the correct location of the MQ JAR files.

2.1.1 Installing UFM as a Windows Service

2.1.1.1 Prerequisites for WebSphere MQ v7.0, v7.1 and V7.5

During the install of IBM MQ, the Java MQ JAR files are installed into the <MQ_Install_Path>\ java\lib\ directory. E.g. ***C:\Program File\IBM\IBM MQ\java\lib***

The user needs to copy the following 6 Java MQ JAR files to the UFM "libs" directory (e.g. ***C:\ Capitalware\UFM\libs***):

1. connector.jar
2. com.ibm.mq.jar
3. com.ibm.mq.commonservices.jar
4. com.ibm.mq.headers.jar
5. com.ibm.mq.jmqi.jar
6. com.ibm.mq.pcf.jar

2.1.1.2 Prerequisites for IBM MQ v8.0, v9.0, v9.1 and higher

During the install of IBM MQ, the Java MQ JAR files are installed into the <MQ_Install_Path>\java\lib\ directory. E.g. ***C:\Program File\IBM\IBM MQ\java\lib***

The user needs to copy the following 1 Java MQ JAR files to the UFM "libs" directory (e.g. ***C:\Capitalware\UFM\libs***):

- com.ibm.mq.allclient.jar

2.1.1.3 Installing the UFM Service

To install UFM as a Windows service, do the following commands from a Command Prompt:

```
cd \Capitalware\UFM
ufm_svc.exe /install
```

The default Windows service name for UFM is "UFMService". To install UFM as many Windows services, do the following commands from a Command Prompt:

```
cd \Capitalware\UFM
ufm_svc.exe /install  UFMService2
ufm_svc.exe /install  UFMService3
```

Note: If you receive the error "***Could not open SCManager***" or "***Access is denied***" (Due to the User Access Control (UAC) feature in Windows), the user will need to open the Command Prompt by right clicking on the application icon and select "Run as an administrator".

2.1.2 Starting or Stopping the UFM Service

To start or stop the UFM Service, the user can use the "net" command from a Command Prompt:

```
net start UFMService
net stop  UFMService
```

The user can also start and stop the UFM Service from the Services MMC console found in the Administrative Tools window.

The default UFM Workflow XML file that the UFM Windows service will use is called: "***ufm.xml***". To use a differently named UFM Workflow XML file for the service, edit the ***service.properties*** file and change the UFM Workflow XML file name.

Example:

```
UFMService = ufm_watch.xml
UFMService2 = ufm_watch2.xml
```

Note: If you receive the error "***Error 1067: The process terminated unexpectedly***", that means the UFM service cannot find a JVM (Java) on your computer. Update your system's ***PATH*** environment variable to point to the JVM bin directory or set the ***JAVA_HOME*** environment variable so that the UFM service can find the JVM.

2.1.3 Uninstalling the UFM Service

To uninstall UFM Service, do the following commands from a Command Prompt:

```
cd \Capitalware\UFM
ufm_svc.exe /uninstall
```

To uninstall a UFM Windows service not using the default name, do the following:

```
cd \Capitalware\UFM
ufm_svc.exe /uninstall UFMService2
ufm_svc.exe /uninstall UFMService3
```

2.2 Unix and Linux Installation

To install Universal File Mover on Unix or Linux, do the following:

1. ftp or copy the selected ufm.tar.zip file to the target platform
2. Unzip and un-tar the ufm.tar.zip to an appropriate directory with the following commands:

```
unzip ufm.tar.zip
tar -xvf ufm.tar
```

3. Change directory to ***Capitalware/UFM/***
4. Next, do the following command:

```
chmod +x *.sh
```

Starting with IBM MQ v7.0.1.8, the MQ software supports multiple installations on Unix and/or Linux servers. Hence, the version of the MQ JAR files the user wishes to use may not be in the default location as in previous version of MQ. The ***UFM.sh*** script will check for the default location and where or not the ***MQ_JAVA_LIB_PATH*** environment variable has been set. To make sure that UFM is using the correct version of the MQ JAR files, simply set the ***MQ_JAVA_LIB_PATH*** environment variable to the correct location or update the ***UFM.bat*** script.

2.3 IBM i Installation

To install Universal File Mover on IBM i (OS/400), do the following:

1. ftp or copy the selected ufm.tar.zip file to the target platform
2. Unzip and un-tar the ufm.tar.zip to an appropriate directory with the following commands:

```
unzip ufm.tar.zip
tar -xvf ufm.tar
```

3. Change directory to ***Capitalware/UFM/***
4. Next, do the following command:

```
chmod +x *.sh
```

3 Starting UFM

This section describes the various ways that the UFM can be started. Currently, UFM can be started via IBM MQ's triggering system, manually, by a scheduling package or a Windows service.

UFM supports process locking, so that the user can make sure that only one instance of a UFM Workflow XML file is being processed at a time (lock_flag of "true") or to make sure another instance of UFM is already running (lock_flag of "false").

3.1 Command-line parameters

This section describes the required and optional command-line parameters for the UFM.

Required Parameter	Description
ufm_workflow.xml	UFM Workflow XML file to be used for this execution.

Optional Parameter	Description
process_lock_filename	The full path and filename of a file that will be used as a process lock (make it a unique name).
lock_flag	Use either "true" or "false". • "true" means only allow UFM to run if the process lock filename does not exist • "false" means only allow UFM to run if the process lock filename does exist

3.2 Triggered Execution

If the local input queue has be setup for triggering (i.e. Trigger First) then when a message arrives on the input queue and the message count goes from 0 to 1 (trigger on first), the queue manager will start UFM.

3.3 Manual Execution

UFM can be manually started from a Command Prompt or Unix shell. First change to the UFM directory (i.e. C:\Capitalware\UFM) and then type the following:

```
ufm.bat sample1.xml
```

3.4 Scheduled Execution

UFM can be setup to be run under a scheduling package. (For example, Windows **Scheduled Task** service or Unix/Linux crontab). Use the scheduling package's tool to setup the necessary parameters for the scheduled execution.

3.5 Windows Service

Please see section 2.1.2 for the details.

3.6 Sample UFM Workflow XML Files

3.6.1 Sample #1

Sample UFM Workflow XML file:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_Workflow SYSTEM "UFM_Workflow.dtd">
<UFM_Workflow>
  <Actions>

    <Append file="data\atest.txt">
      <File dir="data">test.txt</File>
    </Append>

    <MQSend delete="Y">
      <File>data\atest.txt</File>
      <MQ>
        <QMgrName>MQWT1</QMgrName>
        <QueueName>TEST.Q1</QueueName>
      </MQ>
      <Remote>
        <Directory>C:\temp</Directory>
      </Remote>
    </MQSend>

    <MQSend format="S">
      <File>data\abc.txt</File>
      <MQ>
        <QMgrName>MQWT1</QMgrName>
        <QueueName>TEST.Q2</QueueName>
      </MQ>
      <Remote>
        <Directory>C:\temp</Directory>
      </Remote>
    </MQSend>

  </Actions>
</UFM_Workflow>
```

This sample file depicts an UFM Workflow XML file with 3 actions:

1. The Append action appends test.txt file to appendtest.txt
2. The MQSend action sends file atest.txt as a message via MQ and then deletes the file
3. The last MQSend action sends file abc.txt as a message via MQ and marks the MQMD.Format field as 'MQSTR' (string).

3.6.1.1 Running Sample #1 on Windows

```
ufm.bat sample1.xml
```

3.6.1.2 Running Sample #1 on Linux/Unix/IBM i

```
ufm.sh sample1.xml
```

3.6.2 Sample #2

Sample UFM Workflow XML file:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_Workflow SYSTEM "UFM_Workflow.dtd">
<UFM_Workflow>

  <Global>
    <MQ>
      <QMgrName>MQWT1</QMgrName>
      <QueueName>TEST.Q1</QueueName>
    </MQ>
  </Global>

  <Actions>

    <Copy todir=”D:\appdata”>
      <File dir="data">test.txt</File>
    </Copy>

    <MQSend delete="Y">
      <File dir=”D:\appdata”>test.txt</File>
      <Remote>
        <Directory>C:\temp</Directory>
      </Remote>
    </MQSend>

  </Actions>
</UFM_Workflow>
```

This sample file depicts an UFM Workflow XML file with a Global section and 2 actions:

Global

- The Global element contains a <MQ> element that holds the MQ values to be used for all actions in this UFM Workflow XML file.

Actions

1. The Copy action copies test.txt file to D:\appdata\abcxyz.txt
2. The MQSend action sends file test.txt as a message via MQ and then deletes the file

3.6.2.1 Running Sample #2 on Windows

```
ufm.bat sample2.xml
```

3.6.2.2 Running Sample #2 on Linux/Unix/IBM i

```
ufm.sh sample2.xml
```

3.6.3 Sample #3

Sample UFM Workflow XML file:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_Workflow SYSTEM "UFM_Workflow.dtd">
<UFM_Workflow>

  <Global>
    <MQ>
      <MQFile>mq_mqwt1.xml</MQFile>
    </MQ>
  </Global>

  <Actions>

    <Move tofile=”abcxyz.txt” todir=”D:\appdata”>
      <File dir="data">test.txt</File>
    </Move>

    <MQSend delete="Y">
      <File dir=”D:\appdata”>abcxyz.txt</File>
      <Remote>
        <Directory>C:\temp</Directory>
      </Remote>
    </MQSend>

  </Actions>
</UFM_Workflow>
```

This sample file depicts an UFM Workflow XML file with a Global section and 2 actions:

Global

- The Global element contains a <MQ> element which contains a <MQFile> element. The <MQFile> contains the name of a MQ XML file that holds the MQ values to be used for all actions in this UFM Workflow XML file.

Actions

1. The Move action moves the test.txt file to D:\appdata\abcxyz.txt
2. The MQSend action sends file abcxyz.txt as a message via MQ and then deletes the file

3.6.3.1 Running Sample #3 on Windows

```
ufm.bat sample3.xml
```

3.6.3.2 Running Sample #3 on Linux/Unix/IBM i

```
ufm.sh sample3.xml
```

3.6.4 Sample #4

Sample UFM Workflow XML file:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_Workflow SYSTEM "UFM_Workflow.dtd">
<UFM_Workflow>

  <Global>
    <MQ>
      <CCDTFile>C:\tables\MQWT1.TAB</CCDTFile>
      <QMgrName>MQWT1</QMgrName>
      <QueueName>TEST.Q1</QueueName>
    </MQ>
  </Global>

  <Actions>

    <MQSend format="S">
      <File dir="D:\appdata">abc*</File>
      <Remote>
        <Directory>C:\temp</Directory>
      </Remote>
    </MQSend>

    <Execute xmlfile="appjob.xml" />

  </Actions>
</UFM_Workflow>
```

This sample file depicts an UFM Workflow XML file with a Global section and 2 actions:

Global

- The Global element contains a <MQ> element that has a <CCDTFile> element. The <CCDTFile> value is the complete path and filename of a CCDT (Client Channel Definition Table) file.

Actions

1. The MQSend action sends all files that begin with abc* as individual messages via MQ and marks the MQMD.Format field as 'MQSTR' (string).
2. The Execute runs an external program as given in the "appjob.xml" file.

3.6.4.1 Running Sample #4 on Windows

```
ufm.bat sample4.xml
```

3.6.4.2 Running Sample #4 on Linux/Unix/IBM i

```
ufm.sh sample4.xml
```

3.6.5 Sample #5

Sample UFM Workflow XML file:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_Workflow SYSTEM "UFM_Workflow.dtd">
<UFM_Workflow>

  <Global>
    <Property name="app_dir" value="D:\appdata" />
    <Property name="myfile" value="test.txt" />
    <MQ>
      <MQFile>mq_mqwt1.xml</MQFile>
    </MQ>
  </Global>

  <Actions>
    <Move todir="${app_dir}">
      <File dir="data">${myfile}</File>
    </Move>

    <MQSend format="S">
      <File dir="${app_dir}">${myfile}</File>
      <Remote>
        <Directory>C:\temp</Directory>
      </Remote>
    </MQSend>

    <Execute xmlfile="appjob.xml" />

    <Delete>
      <File dir="${app_dir}">${myfile}</File>
    </Delete>
  </Actions>
</UFM_Workflow>
```

This sample file depicts an UFM Workflow XML file with a Global section and 4 actions:

Global

- The Global element contains 3 different user-defined Property tokens followed by a <MQ> element which contains an <MQFile> element.

Actions

1. The Move action moves a file from one directory to another directory.
2. The MQSend action sends a file via MQ and marks the MQMD.Format field as 'MQSTR'.
3. The Execute action runs an external program as given in the “appjob.xml” file.
4. The Delete action deletes the specified file.

3.6.5.1 Running Sample #5 on Windows

```
ufm.bat sample5.xml
```

3.6.5.2 Running Sample #5 on Linux/Unix/IBM i

```
ufm.sh sample5.xml
```

3.6.6 Sample #6

Sample UFM Workflow XML file:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_Workflow SYSTEM "UFM_Workflow.dtd">
<UFM_Workflow>
  <Actions>
    <Watch pollinterval="5">

      <WatchItem type="D">
        <Directory>C:\temp\other</Directory>
        <Extension>txt</Extension>
        <Actions>
          <MQSend delete="Y" format="S">
            <File>${WATCH_FOUND_FILE}</File>
            <MQ>
              <MQFile>mq.xml</MQFile>
            </MQ>
            <Remote>
              <Directory>X:\something</Directory>
            </Remote>
          </MQSend>
        </Actions>
      </WatchItem>

    </Watch>

  </Actions>
</UFM_Workflow>
```

This sample file depicts an UFM Workflow XML file that monitors a directory for files (with 'txt' extension) and when the matching file(s) appears, UFM will execute a series of user-defined Actions.

Actions

1. The Watch action monitors the C:\temp\other\ directory for incoming files when a file appears execute the actions in the Action element
    A. Execute MQSend Action to send the matching file and use the MQ information from the mq.xml file.

3.6.6.1 Running Sample #6 on Windows

```
ufm.bat sample6.xml
```

3.6.6.2 Running Sample #6 on Linux/Unix/IBM i

```
ufm.sh sample6.xml
```

3.6.7 Sample #7

Sample UFM Workflow XML file:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_Workflow SYSTEM "UFM_Workflow.dtd">
<UFM_Workflow>

  <Actions>

    <MQReceive getwithconvert="Y" run="D">
      <MQ>
        <MQFile>mq_mqwt1.xml</MQFile>
        <BackOutQName>TEST.Q1.BK</BackOutQName>
      </MQ>

      <Default>
         <Directory override="Y">C:\temp\UFM</Directory>
         <FileName override="Y">${INDEX}.txt</FileName>
      </Default>

      <Actions>
        <SortUnique todir="C:\temp\UFM" tofile="su_test.txt"
order="A" column="4">
          <File>${MQ_RECEIVE_FILE}</File>
        </SortUnique>

        <MQSend delete="Y" format="S">
          <File dir="C:\temp\UFM">su_test.txt</File>
          <MQ>
            <QMgrName>${MQ_REPLY_TO_QMGR_NAME}</QMgrName>
            <QueueName>${MQ_REPLY_TO_QUEUE_NAME}</QueueName>
          </MQ>
          <Remote>
            <Directory>C:\temp</Directory>
          </Remote>
        </MQSend>

      </Actions>
    </MQReceive>

  </Actions>
</UFM_Workflow>
```

This sample file depicts an UFM Workflow XML file that monitors a directory for files (with 'txt' extension) and when the matching file(s) appears, UFM will execute a series of user-defined Actions.

Actions

1. The MQReceive action runs as a daemon and retrieves messages from the queue when a message is received, execute the actions in the Action element
    A. Execute SortUnique to sort and remove duplicates from the file.
    B. Execute MQSend Action to send a file

3.6.7.1 Running Sample #7 on Windows

```
ufm.bat sample7.xml
```

3.6.7.2 Running Sample #7 on Linux/Unix/IBM i

```
ufm.sh sample7.xml
```

4 UFM_Workflow XML File

The UFM Workflow XML file is a scripting workflow XML file. It is comprised of 2 major elements: ***Global*** and ***Actions***. The Global section defines global values to be used while UFM is running. The Actions section contains Action commands that manipulate files (Append, Copy, Delete, Move, Rename and Zip) and Action commands that move files in and out of MQ (MQSend and MQReceive).

4.1 <UFM_Workflow> Root Element

<UFM_Workflow> is the root element for the UFM Workflow XML file.

4.2 <Global> Element

The Global element (optional) contains 6 elements: Property, Log4JFile, LogFile, OnError, MQ and Status.

Attributes

- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if an Action fails. The default value is 'Y'. This is a global value that can be overridden for each Action.
- **usefilelocking** – [Y / N] – specifies whether or not the file locking will be used with the actions in this Workflow. The default value is 'Y'. This is a global value.

4.2.1 <Property> Element

The Property element(s) (optional) specifies a used-defined token (keyword) and its value. After the property token is defined, it can be used in Action commands as a replaceable token.

Attributes:

- **name** – The name of the user-defined token
- **value** – The value for the user-defined token

Elements:

None.

Example: Two user-defined properties along with a MQSend Action.

```
<Global onerrorfail="Y">
  <Property name="testfile" value="abc.txt" />
  <Property name="temp" value="/tmp" />
</Global>
<Actions>
  <MQSend format="S">
    <File dir="${temp}">${testfile}</File>
    <MQ>
      <QMgrName>MQWT1</QMgrName>
      <QueueName>TEST.Q1</QueueName>
    </MQ>
    <Remote>
      <Directory>C:\temp</Directory>
    </Remote>
  </MQSend>
</Actions>
```

Example: A user-defined Property can have a value that is made up of other UFM tokens.

```
<Global>
  <Property name="testfile" value="abc_${DATE}.txt" />
</Global>
```

Example: A user-defined Property can be used within an Execute Action.

```
<Global>
  <Property name="mymsg"
            value="${DATETIME} There is a problem. Help!" />
</Global>
<Actions>
  <Execute xmlfile="abcrun.xml" />
</Actions>
```

And in the ***abcrun.xml*** Job XML file, the file's contents could be as follows:

```
<UFM_Job>
  <Job name="abcrun">
    <Command wait='y'>C:\pgms\sendalert.exe</Command>
    <Parm>-m</Parm>
    <Parm>${HOSTNAME} - ${mymsg}</Parm>
  </Job>
</UFM_Job>
```

4.2.2 <Log4JFile> Element

The Log4JFile element (optional) specifies the path and filename of the log4j property file. By default, UFM will use the file name 'log4j.properties'.

```
<Global>
  <Log4JFile>C:\apps\log4j.properties</Log4JFile>
</Global>
```

4.2.3 <LogFile> Element

The LogFile element (optional) specifies the path and filename of the log4j logfile.

```
<Global>
  <LogFile>C:\logs\applA.log</LogFile>
</Global>
```

4.2.4 <OnError> Element

The OnError element (optional) contains 2 element: Execute and SendEmail. OnError can contain either Execute or SendEmail Actions or both.

Elements:

- **<Execute>** (optional) specifies an external program to be executed when the UFM Workflow XML script fails.
- **<SendEmail>** (optional) specifies that an email is to be sent when the UFM Workflow XML script fails.

```
<Global>
  <OnError>
    <Execute xmlfile="alert.xml" />
  </OnError>
</Global>
```

4.2.5 <MQ> Element

The MQ element (optional) contains elements that describe how the action connects to the queue manager.

Elements:

- **<MQFile>** (optional) specifies a MQ XML file
- **<CCDTFile>** (optional) specifies a CCDT file
- **<QMgrName>** (optional) specifies the name of the queue manager
- **<QueueName>** (optional) specifies the name of the queue
  Attributes of <QueueName>:
  - "**qmgrname**" (optional) specifies the name of a remote queue manager
    e.g. <QueueName qmgrname="MQC1">TEST.Q1</QueueName>
- **<BackOutQName>** (optional) specifies the name of the backout queue name
  Attributes of <BackOutQName>:
  - "**qmgrname**" (optional) specifies the name of the queue manager
    e.g. <BackOutQName qmgrname="MQA1">TEST.Q1.BO</BackOutQName>
- **<ReplyToQName>** (optional) specifies the name of the reply to queue name
  Attributes of <ReplyToQName>:
  - "**qmgrname**" (optional) specifies the name of the queue manager
    e.g. <ReplyToQName qmgrname="MQS1">TEST.Q2</ReplyToQName>

```
<Global>
  <MQ>
    <MQFile>mq.xml</MQFile>
  </MQ>
</Global>
```

4.2.6 <Status> Element

The Status element (optional) contains an MQ element. If the Status element is specified then a message will be written to a status queue with the results of each action of the UFM Workflow as the Actions are processed. See ***Section 13*** for defining the Status queue.

Attributes:

- **timetolive** – The length of time, in days, that the status message will reside on the status queue. The default value is 7.

Element:

- **<MQ>** (optional) specifies an external program to be executed when the UFM Workflow XML script fails.

```
<Global>
  <Status>
    <MQ>
      <QMgrName>MQWT1</QMgrName>
      <QueueName>CAPITALWARE.UFM.STATUS</QueueName>
    </MQ>
  </Status>
</Global>
```

Note: It is a good idea to define a single status queue on 1 queue manager then have all UFM Workflows send the status information to a single queue.

4.3 IBM MQ Action Elements

There are 3 IBM MQ related action elements and are detailed below.

4.3.1 <MQSend> Element

This section describes how to invoke the MQSend Action. The MQSend Action will read the contents of a file and place it on a queue as a message. The MQSend Action will include Path and Filename information in the message's MQMD. This information details the file's location on the remote system. The MQSend Action can handle a file of any size and is sent as one or more messages to a queue. MQSend Action supports both string and binary message formats. The MQSend Action can also specify an ***Execute*** Action that will be executed on the remote server. The MQSend Action supports a wildcard (i.e. *.txt) in the filename that will send multiple files as separate messages in a single execution. The MQSend Action can compress and/or encrypt the message data.. Once the data is compressed and/or encrypted, there is no conversion between different platforms (i.e. ASCII to EBCDIC). The MQSend Action can launch a local external program after putting the message on the queue (see section 8 for more details). It can also delete or archive the file after the message has been put on the queue. The MQSend Action can send the same message to multiple queues using the DistributionList element in the UFM_MQ XML file.

Note: US export restrictions limit the Java AES support to 128-bits. If the user wishes to use AES 192 or 256-bit encryption then ***"Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 6"*** needs to be downloaded (and installed) from Oracle's Java web page: http://www.oracle.com/technetwork/java/javase/downloads/index.html

4.3.1.1 Attributes

- **delete** – [Y / N] – deletes the files after it is sent
- **format** – [N / S / Z] – sets the messages MQMD.Format field to None (N), String (S) or compresses (Z) the message data.
- **ccsid** – sets the message's CodedCharSetId. The default is 0 meaning it is set by the queue manager.
- **priority** – [-1 / 0 / 1 / 2 / 3 / 4 / 5 / 6 / 7 / 8 / 9] – sets the priority of the message
- **persistence** – [Y / N] – sets whether or not the message should be persistent
- **keysize** – [128 / 192 / 256] (optional) – specifies the AES key size used for the encryption / decryption of the message data. The default value is 128.
- **passphrase** (optional) – specifies a user supplied PassPhrase.
  Suggested PassPhrase sizes:
  - For KeySize of 128-bits, the PassPhrase should be 16 bytes
  - For KeySize of 192-bits, the PassPhrase should be 24 bytes
  - For KeySize of 256-bits, the PassPhrase should be 32 bytes
- **usev7prop** – specifies UFM use MQ properties for Remote FileName, Directory and JobName values. The default value is 'Y'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

```
<Send delete="N" format="N" persistence="Y" priority="-1">

<Send delete="N" format="N" persistence="Y" priority="-1"
keysize="128" passphrase="this is a secret">
```

4.3.1.2 Elements

- **<File>** (required) specifies the name of the file to be sent
  Attributes:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

- **<MQ>** (required) element contains elements that describe how the MQSend Action is to connect to the queue manager.
  - **<MQFile>** (optional) specifies a MQ XML file
  - **<CCDTFile>** (optional) specifies a CCDT file
  - **<QMgrName>** (optional) specifies the name of the queue manager
  - **<QueueName>** (optional) specifies the name of the queue
    Attributes of <QueueName>:
    - "**qmgrname**" (optional) specifies the name of a remote queue manager
      e.g. <QueueName qmgrname="MQC1">TEST.Q1</QueueName>
  - **<ReplyToQName>** (optional) specifies the name of the reply to queue name
    Attributes of <ReplyToQName>:
    - "**qmgrname**" (optional) specifies the name of the queue manager
      e.g. <ReplyToQName qmgrname="MQS1">TEST.Q2</ReplyToQName>

- **<Execute>** (optional) specifies a local external program to be executed

- **<Archive>** (optional) specifies the directory and filename where the file will be moved to after the file is sent
  Attributes:
  - "todir" (optional) specifies the directory where the file is located
  - "tofile" (optional) specifies the name of the archive file

  e.g. <Archive todir="C:\test" tofile="saved.txt" />

- **<Remote>** (optional) element contains elements that will specify values to be used by the receiving component of UFM
  - **<Execute>** (optional) specifies a remote external program to be executed
  - **<Directory>** (optional) specifies the remote Job XML file that defines the external program to be executed
  - **<FileName>** (optional) specifies the filename to be used on the remote server

4.3.1.3 Example

An example of a MQSend Action:

```
<MQSend delete="N" format="N">
  <File>data\test.xml</File>
  <MQ>
    <QMgrName>MQWT1</QMgrName>
    <QueueName>TEST.Q1</QueueName>
  </MQ>
  <Remote>
    <Directory>C:\temp</Directory>
  </Remote>
</MQSend>
```

4.3.2 <MQReceive> Element

This section describes how to invoke the MQReceive Action. The MQReceive Action will retrieve a message from a queue and write it to a file. The MQReceive Action can be run in 1 of 4 modes: one-time, triggered, as a daemon or as a Windows service. The MQReceive Action retrieves the message and writes it to a file under a Unit of Work (UOW).

If the incoming message is not a UFM message and the user did not specify the Default File name, the Receive Action will use the following format for the file name:
UFM_####.txt (where #### is the message count number).

If the MQReceive Action detects a compressed and/or encrypted incoming message, it will decompress and/or decrypt the message before writing it to the output file.

If the incoming message has a specified ***Execute*** Action, the MQReceive Action will launch an external program after retrieving the message from the queue (see section 8 for more details). The user can assign a default Execute to the Receive Action, so that an external program will be launched when a message is retrieved.

If the MQReceive Action has a problem with a message (invalid filename or invalid directory), the message will be backed out and the MQReceive Action will abnormally terminate. In order to avoid termination when a problem message is encountered, the user needs to specify a backout queue name (***BackOutQName***). As a result, the MQReceive Action will move the problem message to the backout queue and process the next message.

For the MQReceive Action to exit gracefully while running as a daemon, the MQPutQuit component is used to place a 'Quit' message on the queue.

Note: US export restrictions limit the Java AES support to 128-bits. If the user wishes to use AES 192 or 256-bit encryption then ***"Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 6"*** needs to be downloaded (and installed) from Oracle's Java web page: http://www.oracle.com/technetwork/java/javase/downloads/index.html

4.3.2.1 Attributes

- **append** [Y / N] (optional) specifies whether or not the data will be appended to the target file. The default value is 'N'.
- **getwithconvert** – [Y / N] – If "Y", the Receive Action will issue a MQGet API call with convert option enabled
- **run** – [S / E / D] – specifies 1 of 3 modes: Single "S", Empty "E" and Daemon "D".
- **keysize** – [128 / 192 / 256] (optional) – specifies the AES key size used for the encryption / decryption of the message data. The default value is 128.
- **passphrase** (optional) – specifies a user supplied PassPhrase.
  Suggested PassPhrase sizes:
  - For KeySize of 128-bits, the PassPhrase should be 16 bytes
  - For KeySize of 192-bits, the PassPhrase should be 24 bytes
  - For KeySize of 256-bits, the PassPhrase should be 32 bytes
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

```
<MQReceive getwithconvert="N" run="E">

<MQReceive getwithconvert="N" run="E" keysize="128" passphrase="this
is a secret">
```

4.3.2.2 Elements

- **<MQ>** (required) element contains elements that describe how the action is to connect to the queue manager.
  - **<MQFile>** (optional) specifies a MQ XML file
  - **<CCDTFile>** (optional) specifies a CCDT file
  - **<QMgrName>** (optional) specifies the name of the queue manager
  - **<QueueName>** (optional) specifies the name of the queue
  - **<BackOutQName>** (optional) specifies the name of the backout queue name
    Attributes of <BackOutQName>:
    - "**qmgrname**" (optional) specifies the name of the queue manager
      e.g. <BackOutQName qmgrname="MQA1">TEST.Q1.BO</BackOutQName>

- **<Default>** (optional) element contains elements that will specify values to be used by the receiving component of UFM
  - **<Execute>** (optional) specifies a local external program to be executed
  - **<Directory>** (optional) specifies the local Job XML file that defines the external program to be executed
  - **<FileName>** (optional) specifies the local filename to be used

- **<Actions>** (optional) specifies a list of Actions to be executed. Any valid Action can be contained in the list of Actions.

4.3.2.3 Example

An example of a MQReceive Action:

```
<MQReceive getwithconvert="N" run="E">
  <MQ>
    <QMgrName>MQWT1</QMgrName>
    <QueueName>TEST.Q1</QueueName>
    <BackOutQName>TEST.Q1.BKOUT</BackOutQName>
  </MQ>
</MQReceive>
```

An example of a MQReceive Action with embedded Actions:

```
<MQReceive getwithconvert="Y" run="D">
  <MQ>
    <MQFile>mq.xml</MQFile>
    <QueueName>TEST.Q1</QueueName>
    <BackOutQName>TEST.Q1.BKOUT</BackOutQName>
  </MQ>

  <Default>
     <Directory override="Y">C:\temp\UFM</Directory>
     <FileName override="Y">${INDEX}.txt</FileName>
  </Default>

  <Actions>
    <SortUnique todir="C:\temp\UFM" tofile="sort_uniq_test.txt"
order="A" column="4">
      <File>${MQ_RECEIVE_FILE}</File>
    </SortUnique>

    <MQSend delete="Y" format="S">
      <File dir="C:\temp\UFM">sort_uniq_test.txt</File>
      <MQ>
        <QMgrName>${MQ_REPLY_TO_QMGR_NAME}</QMgrName>
        <QueueName>${MQ_REPLY_TO_QUEUE_NAME}</QueueName>
      </MQ>
      <Remote>
        <Directory>C:\temp</Directory>
      </Remote>
    </MQSend>
  </Actions>

</MQReceive>
```

Notes:

- The ${MQ_RECEIVE_FILE} variable contains the path and name of the file that was used by the MQReceive action to write the message data to.
- The ${MQ_REPLY_TO_QMGR_NAME} variable contains the Reply-To queue manager name of the incoming message.
- The ${MQ_REPLY_TO_QUEUE_NAME} variable contains the Reply-To queue name of the incoming message.

4.3.3 <MQPutQuit> Element

This section describes how to invoke the MQPutQuit Action. The MQPutQuit Action is used to place a special 'QUIT' message on a queue. MQPutQuit is used to stop the Receive Action component when it is running as a daemon.

4.3.3.1 Attributes

- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.3.3.2 Elements

- **<MQ>** (required) element contains elements that describe how the action is to connect to the queue manager.
  - **<MQFile>** (optional) specifies a MQ XML file
  - **<CCDTFile>** (optional) specifies a CCDT file
  - **<QMgrName>** (optional) specifies the name of the queue manager
  - **<QueueName>** (optional) specifies the name of the queue
    Attributes of <QueueName>:
    - "**qmgrname**" (optional) specifies the name of a remote queue manager
      e.g. <QueueName qmgrname="MQC1">TEST.Q1</QueueName>

4.3.3.3 Example

An example of a MQPutQuit Action:

```
<MQPutQuit>
  <MQ>
    <QMgrName>MQWT1</QMgrName>
    <QueueName>TEST.Q1</QueueName>
  </MQ>
</MQPutQuit>
```

4.4 Network Action Elements

There are 6 network related action elements and are detailed below.

4.4.1 <Ftp> Element

This section describes how to invoke the Ftp Action. The Ftp Action is used to get and/or put files using FTP (File Transfer Protocol) from/to one host to another host over a TCP-based network.

4.4.1.1 Attributes

- **xmlfile** (required) specifies the name of the FTP XML file (see section 6 for more details)
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.4.1.2 Elements

None

4.4.1.3 Example

An example of a Ftp Action:

```
<Ftp xmlfile="ftp1.xml" />
```

4.4.2 <HttpGetFile> Element

This section describes how to invoke the HttpGetFile Action. The HttpGetFile Action is used to get a file from a web server using HTTP (Hypertext Transfer Protocol).

4.4.2.1 Attributes

- **url** (required) specifies the URL (Uniform Resource Locator) of the file to be retrieved via HTTP
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.4.2.2 Elements

- **<File>** (required) specifies the name of the source file
  Attributes of <File>:
- "dir" - specifies the directory where the file is located
  e.g. <File dir="C:\test">data.txt</File>

4.4.2.3 Example

An example of a HttpGetFile Action:

```
<HttpGetFile url="http://www.acme.com/test1.txt">
  <File dir="C:\temp\UFM">test1.txt</File>
</HttpGetFile>
```

4.4.3 <HttpPutFile> Element

This section describes how to invoke the HttpPutFile Action. The HttpPutFile Action is used to store a file on a web server using HTTP (Hypertext Transfer Protocol).

4.4.3.1 Attributes

- **url** (required) specifies the URL (Uniform Resource Locator) of the file to be retrieved via HTTP
- **formdataname** – specifies a 'form-data name' for the 'Content-Disposition' HTTP attribute. The default value is 'upload'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.4.3.2 Elements

- **<File>** (required) specifies the name of the source file
  Attributes of <File>:
- "dir" - specifies the directory where the file is located
  e.g. <File dir="C:\test">data.txt</File>

4.4.3.3 Example

An example of a HttpPutFile Action:

```
<HttpPutFile url="http://www.acme.com/test1.txt">
  <File dir="C:\temp\UFM">test1.txt</File>
</HttpPutFile>
```

4.4.4 <Scp> Element

This section describes how to invoke the Scp Action. The Scp Action is used securely copy file to/from one host to another host.

4.4.4.1 Attributes

- **xmlfile** (required) specifies the name of the SCP XML file (see section 10 for more details)
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.4.4.2 Elements

None

4.4.4.3 Example

An example of a Scp Action:

```
<Scp xmlfile="scp1.xml" />
```

4.4.5 <SendEmail> Element

This section describes how to invoke the SendEmail Action. The SendEmail Action is used to send an email via SMTP (Simple Mail Transfer Protocol) to 1 or more recipients.

4.4.5.1 Attributes

- **xmlfile** (required) specifies the name of the Email XML file (see section 5 for more details)
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.4.5.2 Elements

None

4.4.5.3 Example

An example of an SendEmail Action:

```
<SendEmail xmlfile="sendemail.xml" />
```

4.4.6 <SFtp> Element

This section describes how to invoke the SFtp Action. The SFtp Action is used to securely get and/or put files using SSH FTP (File Transfer Protocol) from/to one host to another host over a TCP-based network.

4.4.6.1 Attributes

- **xmlfile** (required) specifies the name of the SFTP XML file (see section 11 for more details)
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.4.6.2 Elements

None

4.4.6.3 Example

An example of a SFtp Action:

```
<SFtp xmlfile="sftp1.xml" />
```

4.5 File Action Elements

There are 24 file related action elements and are detailed below.

4.5.1 <Append> Element

This section describes how to invoke the Append Action. The Append Action is used to append one file to another file.

4.5.1.1 Attributes

- **todir** (optional) specifies the target directory
- **tofile** (required) specifies the name of the target file
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.1.2 Elements

- **<File>** (required) specifies the name of the source file
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.1.3 Example

An example of an Append Action:

```
<Append tofile="appendtest.txt">
  <File dir="data">test.xml</File>
</Append>
```

4.5.2 <Convert> Element

This section describes how to invoke the Convert Action. The Convert Action is used to convert file file from one OS (operating system) format to another format. The Convert Action can convert files between: DOS, EBCDIC, Mac OS X and UNIX/Linux.

4.5.2.1 Attributes

- **todir** (required) specifies the target directory
- **tofile** (optional) specifies a new target file name
- **mode** (required) specifies the type of conversion to take place. Valid values are:
    - **D2E** - DOS to EBCDIC conversion
    - **D2M** - DOS to Mac OS X conversion
    - **D2U** - DOS to UNIX/Linux conversion
    - **E2D** - EBCDIC to DOS conversion
    - **E2M** - EBCDIC to Mac OS X conversion
    - **E2U** - EBCDIC to UNIX/Linux conversion
    - **M2D** - Mac OS X to DOS conversion
    - **M2E** - Mac OS X to EBCDIC conversion
    - **M2U** - Mac OS X to UNIX/Linux conversion
    - **U2D** - UNIX/Linux to DOS conversion
    - **U2E** - UNIX/Linux to EBCDIC conversion
    - **U2M** - UNIX/Linux to Mac OS X conversion
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.2.2 Elements

- **<File>** (required) specifies the name of the file to be copied
  Attributes of <File>:
    - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.2.3 Example

An example of a Convert Action:

```
<Convert mode="D2U" todir="C:\temp" tofile="test_unix.xml">
  <File dir="data">test.xml</File>
</Convert>
```

4.5.3 <Copy> Element

This section describes how to invoke the Copy Action. The Copy Action is used to copy one or more files from one directory to another directory.

4.5.3.1 Attributes

- **todir** (required) specifies the target directory where the file is to be copied to
- **tofile** (optional) specifies a new target file name
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.3.2 Elements

- **<File>** (required) specifies the name of the file to be copied
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.3.3 Example

An example of a Copy Action:

```
<Copy todir="C:\temp\">
  <File dir="data">test.xml</File>
</Copy>
```

4.5.4 <DecryptFile> Element

This section describes how to invoke the DecryptFile Action. The DecryptFile Action is used to decrypt an encrypted file to another file.

Note: US export restrictions limit the Java AES support to 128-bits. If the user wishes to use AES 192 or 256-bit encryption then ***"Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 6"*** needs to be downloaded (and installed) from Oracle's Java web page: http://www.oracle.com/technetwork/java/javase/downloads/index.html

4.5.4.1 Attributes

- **todir** (optional) specifies the target directory
- **tofile** (required) specifies the name of the target file
- **keysize** – [128 / 192 / 256] (required) – specifies the AES key size used for the encryption / decryption of the message data. The default value is 128.
- **passphrase** (required) – specifies a user supplied PassPhrase.
  Suggested PassPhrase sizes:
  - For KeySize of 128-bits, the PassPhrase should be 16 bytes
  - For KeySize of 192-bits, the PassPhrase should be 24 bytes
  - For KeySize of 256-bits, the PassPhrase should be 32 bytes
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.4.2 Elements

- **<File>** (required) specifies the name of the source/encrypted file
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.4.3 Example

An example of an DecryptFile Action:

```
<DecryptFile tofile="test.txt" keysize="128" passphrase="this is a secret">
  <File dir="data">test.enc</File>
</DecryptFile>
```

4.5.5 <Delete> Element

This section describes how to invoke the Delete Action. The Delete Action is used to delete one or more files.

4.5.5.1 Attributes

- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.5.2 Elements

- **<File>** (required) specifies the name of the file to be deleted
  Attributes of <File>:
  - "dir" (optional) specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.5.3 Example

An example of a Delete Action:

```
<Delete>
  <File dir="data">test.xml</File>
</Delete>
```

Another example of a Delete Action:

```
<Delete>
  <File dir="data">test.xml</File>
  <File dir="data">test2.xml</File>
  <File dir="data">test3.xml</File>
</Delete>
```

4.5.6 <EncryptFile> Element

This section describes how to invoke the EncryptFile Action. The EncryptFile Action is used to encrypt a file to another file.

Note: US export restrictions limit the Java AES support to 128-bits. If the user wishes to use AES 192 or 256-bit encryption then ***"Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 6"*** needs to be downloaded (and installed) from Oracle's Java web page: http://www.oracle.com/technetwork/java/javase/downloads/index.html

4.5.6.1 Attributes

- **todir** (optional) specifies the target directory
- **tofile** (required) specifies the name of the target file
- **keysize** – [128 / 192 / 256] (required) – specifies the AES key size used for the encryption / decryption of the message data. The default value is 128.
- **passphrase** (required) – specifies a user supplied PassPhrase.
  Suggested PassPhrase sizes:
  - For KeySize of 128-bits, the PassPhrase should be 16 bytes
  - For KeySize of 192-bits, the PassPhrase should be 24 bytes
  - For KeySize of 256-bits, the PassPhrase should be 32 bytes
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.6.2 Elements

- **<File>** (required) specifies the name of the source file
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.6.3 Example

An example of an EncryptFile Action:

```
<EncryptFile tofile="test.enc" keysize="128" passphrase="this is a secret">
  <File dir="data">test.txt</File>
</EncryptFile>
```

4.5.7 <MakeDir> Element

This section describes how to invoke the MakeDir Action. The MakeDir Action is used to create a directory.

4.5.7.1 Attributes

- **dir** (required) specifies the target directory
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.7.2 Elements

None.

4.5.7.3 Example

An example of a MakeDir Action:

```
<MakeDir dir="C:\temp\new" />
```

4.5.8 <Merge> Element

This section describes how to invoke the Merge Action. The Merge Action is used to merge 2 or more files to another file.

4.5.8.1 Attributes

- **todir** (optional) specifies the target directory
- **tofile** (required) specifies the name of the target file
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.8.2 Elements

The File element repeats for each source file.

- **<File>** (required) specifies the name of one of the source file
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.8.3 Example

An example of an Merge Action:

```
<Merge todir="C:\temp" tofile="mergetest.txt">
  <File dir="data">test.txt</File>
  <File dir="data">test2.txt</File>
  <File dir="data">test3.txt</File>
  <File dir="data">test4.txt</File>
</Merge>
```

4.5.9 <MergeSort> Element

This section describes how to invoke the MergeSort Action. The MergeSort Action is used to merge 2 or more files, sort the data into another file.

4.5.9.1 Attributes

- **todir** (optional) specifies the target directory
- **tofile** (required) specifies the name of the target file
- **column** (optional) specifies the starting column to sort the data on. The default value is "1",
- **order** [A/D] (optional) specifies sort order of either ascending or descending. The default value is "A".
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.9.2 Elements

The File element repeats for each source file.

- **<File>** (required) specifies the name of one of the source file
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.9.3 Example

An example of an MergeSort Action:

```
<MergeSort tofile="mergetest.txt" column="1" order="A">
  <File dir="data">test.txt</File>
  <File dir="data">test2.txt</File>
  <File dir="data">test3.txt</File>
  <File dir="data">test4.txt</File>
</MergeSort>
```

4.5.10 <Move> Element

This section describes how to invoke the Move Action. The Move Action is used to move one or more files from one directory to another directory.

4.5.10.1 Attributes

- **todir** (required) specifies the target directory where the file is to be moved to
- **tofile** (optional) specifies a new target file name
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.10.2 Elements

- **<File>** (required) specifies the name of the file to be moved
  Attributes of <File>:
  - "dir" (optional) specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.10.3 Example

An example of a Move Action:

```
<Move todir="C:\temp\">
  <File dir="data">test.xml</File>
</Move>
```

4.5.11 <ReEncode> Element

This section describes how to invoke the ReEncode Action. The ReEncode Action is used to change the character encoding of a file (i.e. US-ASCII to UTF-16).

4.5.11.1 Attributes

- **todir** (required) specifies the target directory
- **tofile** (optional) specifies a new target file name
- **inputencoding** (required) specifies the input character set of the file. i.e. US-ASCII
- **outputencoding** (required) specifies the output character set for the new file. i.e. UTF-16
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.11.2 Elements

- **<File>** (required) specifies the name of the file to be copied
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.11.3 Example

An example of a ReEncode Action:

```
<ReEncode inputencoding="US-ASCII" outputencoding="UTF-16"
todir="C:\temp" tofile="new_test.xml">
  <File dir="data">test.xml</File>
</ReEncode>
```

4.5.12 <RemoveDir> Element

This section describes how to invoke the RemoveDir Action. The RemoveDir Action is used to delete a directory.

4.5.12.1 Attributes

- **dir** (required) specifies the target directory
- **removeall** [Y / N] (optional) specifies whether or not the files and directories in the target directory are to be deteleted as well. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.12.2 Elements

None.

4.5.12.3 Example

An example of a RemoveDir Action:

```
<RemoveDir dir="C:\temp\new" removeall="N" />
```

4.5.13 <RemoveItems> Element

This section describes how to invoke the RemoveItems Action. The RemoveItems Action is used to remove items in 1 list from items in another list.

4.5.13.1 Attributes

- **todir** (optional) specifies the target directory
- **tofile** (required) specifies the name of the target file
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.13.2 Elements

The File element repeats for each source file.

- **<File>** (required) specifies the name of the source file
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

- **<ItemFile>** (required) specifies the name of file that contains the list of items to be removed from the source file
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">removelist.txt</File>

4.5.13.3 Example

An example of an RemoveItems Action:

```
<RemoveItems todir="C:\temp\UFM">
  <File dir="data">test_list.txt</File>
  <ItemFile dir="data">removelist.txt</ItemFile>
</RemoveItems>
```

4.5.14 <Rename> Element

This section describes how to invoke the Rename Action. The Rename Action is used to rename a file.

4.5.14.1 Attributes

- **tofile** (required) specifies a new file name
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.14.2 Elements

- **<File>** (required) specifies the name of the file to be renamed
  Attributes of <File>:
  - "dir" (optional) specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.14.3 Example

An example of a Rename Action:

```
<Rename tofile="test2.xml">
  <File dir="data">test.xml</File>
</Rename>
```

4.5.15 <ReplaceText> Element

This section describes how to invoke the ReplaceText Action. The ReplaceText Action is used to search and replace text in a file.

4.5.15.1 Attributes

- **todir** (optional) specifies the target directory
- **tofile** (required) specifies a new target file name
- **rows** (required) specifies the rows to perform the search against. The default is "1:*" which is all rows. The layout of the field is S:E where S is the starting row and E is the ending row ("*" until the end of the file)
- **columns** (required) specifies the columns to perform the search against. The default is "1:*" which is all columns. The layout of the field is S:E where S is the starting column and E is the ending column ("*" up to the last column of data)
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.15.2 Elements

- **<File>** (required) specifies the name of the source file
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

- **<Find>** (required) specifies the text to be searched for
  e.g. <Find>XXX</Find>

- **<Replace>** (required) specifies the replacement text
  e.g. <Replace>YYY</Replace>

4.5.15.3 Example

An example of an ReplaceText Action:

```
<ReplaceText tofile="test_new.txt" rows="1:*" columns="1:*">
  <Find>000</Find>
  <Replace>!!!</Replace>
  <File>data\test.txt</File>
</ReplaceText>
```

4.5.16 <Sort> Element

This section describes how to invoke the Sort Action. The Sort Action is used to sort the data into another file.

4.5.16.1 Attributes

- **todir** (optional) specifies the target directory
- **tofile** (required) specifies a new target file name
- **column** (optional) specifies the starting column to sort the data on. The default value is "1",
- **order** [A/D] (optional) specifies sort order of either ascending or descending. The default value is "A".
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.16.2 Elements

The File element repeats for each source file.

- **<File>** (required) specifies the name of one of the source file
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.16.3 Example

An example of a Sort Action:

```
<Sort tofile="mergetest.txt" column="1" order="A">
  <File dir="data">test.txt</File>
</Sort>
```

4.5.17 <SortUnique> Element

This section describes how to invoke the SortUnique Action. The SortUnique Action is used to first sort the data and then collapse adjacent identical lines to one (i.e. remove duplicate lines).

4.5.17.1 Attributes

- **todir** (optional) specifies the target directory
- **tofile** (required) specifies a new target file name
- **column** (optional) specifies the starting column to sort the data on. The default value is "1",
- **order** [A/D] (optional) specifies sort order of either ascending or descending. The default value is "A".
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.17.2 Elements

The File element repeats for each source file.

- **<File>** (required) specifies the name of one of the source file
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.17.3 Example

An example of a SortUnique Action:

```
<SortUnique tofile="test_unique.txt" column="1" order="A">
  <File dir="data">test.txt</File>
</SortUnique>
```

4.5.18 <Split> Element

This section describes how to invoke the Split Action. The Split Action is used to split a file into two or more smaller files.

4.5.18.1 Attributes

- **todir** (required) specifies the target directory
- **type** [B / L] (required) specifies whether the file splitting is to be done by binary (B) or by lines (L). The default value is 'B'.
- **szie** (required) specifies either the number of bytes or the number of lines for the file splitting
- **prefix** (optional) specifies the file prefix for the target file
- **extension** (optional) specifies the file extension for the target file
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.18.2 Elements

The File element repeats for each source file.

- **<File>** (required) specifies the name of one of the source file
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.18.3 Example

An example of a Split Action:

```
<Split type="L" size="10" todir="C:\temp" prefix="text_sm" extension="txt">
  <File>data\test.large.txt</File>
</Split>
```

4.5.19 <Tar> Element

This section describes how to invoke the Tar Action. The Tar Action is used to create a tar archive from the contents of a file or files in a directory.

4.5.19.1 Attributes

- **todir** (optional) specifies the target directory
- **tofile** (required) specifies a new target file name
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.19.2 Elements

- **<File>** (required) specifies the name of the file to be compressed
  Attributes of <File>:
  - "dir" (optional) specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.19.3 Example

An example of a Tar Action:

```
<Tar todir="C:\temp" tofile="test.tar">
  <File dir="data">test.xml</File>
</Tar>
```

4.5.20 <Touch> Element

This section describes how to invoke the Touch Action. The Touch Action is used to set the modification time of the file to the current time of day. If the file doesn't exist, it is created.

4.5.20.1 Attributes

- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.20.2 Elements

- **<File>** (required) specifies the name of the source file
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.20.3 Example

An example of a Touch Action:

```
<Touch>
  <File dir="data">test.xml</File>
</Touch>
```

Another example of an Touch Action:

```
<Touch>
  <File dir="data">test.xml</File>
  <File dir="data">test2.xml</File>
  <File dir="data">test3.xml</File>
</Touch>
```

4.5.21 <Unique> Element

This section describes how to invoke the Unique Action. The Unique Action is used to collapse adjacent identical lines to one (i.e. remove duplicate lines).

4.5.21.1 Attributes

- **todir** (optional) specifies the target directory
- **tofile** (required) specifies a new target file name
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.21.2 Elements

The File element repeats for each source file.

- **<File>** (required) specifies the name of one of the source file
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.21.3 Example

An example of a Unique Action:

```
<Unique tofile="test_unique.txt">
  <File dir="data">test.txt</File>
</Unique>
```

4.5.22 <UnTar> Element

This section describes how to invoke the UnTar Action. The UnTar Action is used to extract the contents of a tar archive to a directory.

4.5.22.1 Attributes

- **todir** (required) specifies a target directory
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.22.2 Elements

- **<File>** (required) specifies the name of the tar file
  Attributes of <File>:
  - "dir" (optional) specifies the directory where the file is located

  e.g. <File dir="C:\test">data.tar</File>

4.5.22.3 Example

An example of a UnTar Action:

```
<UnTar todir="C:\temp">
  <File dir="data">test.tar</File>
</UnTar>
```

4.5.23 <UnZip> Element

This section describes how to invoke the UnZip Action. The UnZip Action is used to extract the contents of a zip archive to a directory.

4.5.23.1 Attributes

- **todir** (required) specifies a target directory
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.23.2 Elements

- **<File>** (required) specifies the name of the zip file
  Attributes of <File>:
  - "dir" (optional) specifies the directory where the file is located

  e.g. <File dir="C:\test">data.zip</File>

4.5.23.3 Example

An example of a Zip Action:

```
<UnZip todir="C:\temp">
  <File dir="data">test.zip</File>
</UnZip>
```

4.5.24 <Watch> Element

This section describes how to invoke the Watch Action. The Watch Action will monitor for a particular file or monitor a directory for a file(s) to appear. The Watch Action can be run in 1 of 3 modes: triggered, as a daemon or as a Windows service. When a matching file appears then UFM will execute a series of user-defined Actions.

4.5.24.1 Attributes

- "**pollinterval**" specifies the polling interval in seconds for this set of Watch elements. The PollInterval value must be larger than 5 seconds.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.24.2 Elements

- **<WatchItem>** element (required) contains the information to monitor the file(s) or files in a directory. WatchItem element will have 2 elements (File and Actions) when the type attribute is set to "F" and up to 3 elements (Directory, Extension (optional) and Actions) when the type attribute is set to "D".

  Attribute of <WatchItem>:

  - "type" [F / D] specifies whether Watch Action is monitoring a file or files within a directory

  Elements of <WatchItem>:

  When Watch element's ***type*** attribute has a value of "F", use the <File> element:

  - **<File>** (required) specifies the name of the file to be moved
    Attributes of <File>:
    - "dir" (optional) specifies the directory where the file is located

    e.g. <File dir="C:\test">data.txt</File>

  When Watch element's ***type*** attribute has a value of "D", use the <Directory> element with or without the <Extension> element:

  - **<Directory>** (required) specifies the complete path to a directory
    Attributes of <Directory>:
    - "sendonstartup" [Y/N] (optional) specifies that UFM should process matching files in the directory. Otherwise, existing files are ignored and only new files are processed.

    e.g. <Directory sendonstartup="Y">C:\work\test</Directory>

  - **<Extension>** (optional) specifies the file extension to monitor in a directory

- **<Actions>** (required) specifies a list of Actions to be executed. Any valid Action can be contained in the list of Actions.

4.5.24.3 Example

An example of a Watch Action:

```
<Watch pollinterval="5">

  <WatchItem type="F">
    <File>C:\temp\amce.txt</File>

    <Actions>
      <MQSend delete="N" format="S">
        <File>${WATCH_FOUND_FILE}</File>
        <MQ>
          <MQFile>mq.xml</MQFile>
        </MQ>
        <Remote>
          <Directory>D:\some\directory</Directory>
        </Remote>
      </MQSend>
    </Actions>
  </WatchItem>

  <WatchItem type="D">
    <Directory>C:\temp\HR</Directory>
    <Extension>txt</Extension>

    <Actions>
      <MQSend delete="Y" format="S">
        <File>${WATCH_FOUND_FILE}</File>
        <MQ>
          <MQFile>mq.xml</MQFile>
        </MQ>
        <Remote>
          <Directory>D:\some_other\directory</Directory>
        </Remote>
      </MQSend>
    </Actions>
  </WatchItem>
</Watch>
```

Note: The ${WATCH_FOUND_FILE} variable is by the user as a place holder to the “found file” from the WatchItem. UFM will transform the variable to the actual file name when the Action is executed.

4.5.25 <WriteText> Element

This section describes how to invoke the WriteText Action. The WriteText Action is used to write/output text to a file.

4.5.25.1 Attributes

- **todir** (optional) specifies the target directory
- **tofile** (required) specifies a new target file name
- **append** [Y / N] (optional) specifies whether or not the data will be appended to the target file. The default value is 'N'.
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.25.2 Elements

The File element repeats for each source file.

- **<File>** (required) specifies the name of one of the source file
  Attributes of <File>:
  - "dir" - specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.25.3 Example

An example of a WriteText Action:

```
<WriteText todir="C:\temp" tofile="text.txt>This is line 1.
This is line 2.
This is line 3.
This is line 4.
</WriteText>
```

4.5.26 <Zip> Element

This section describes how to invoke the Zip Action. The Zip Action is used to compress the contents of a file or files in a directory.

4.5.26.1 Attributes

- **todir** (optional) specifies the target directory
- **tofile** (required) specifies a new target file name
- **createdir** – [Y / N] – specifies whether or not the target directory should be created if no such directory exists. The default value is 'N'.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.5.26.2 Elements

- **<File>** (required) specifies the name of the file to be compressed
  Attributes of <File>:
  - "dir" (optional) specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

4.5.26.3 Example

An example of a Zip Action:

```
<Zip todir="C:\temp" tofile="test.zip">
  <File dir="data">test.xml</File>
</Zip>
```

An example of a Zip Action compressing the entire contents of a directory:

```
<Zip todir="C:\temp" tofile="test.zip">
  <File dir="data" />
</Zip>
```

4.6 Control Action Elements

There are 3 Control action elements and are detailed below.

4.6.1 <If> Element

This section describes how to invoke the If and If/Else Action. The If Action is used to perform a conditional test against an action's variable.

4.6.1.1 Attributes

- **labelname** (optional) – specifies the label name of the action for the conditional test. The default value is “default”.
- **actionname** – specifies the action name for the conditional test.
- **fieldname** – specifies the field name of the action for the conditional test.
- **operator** – specifies the type of conditional test to be performed. Valid values are: EQ, NE, LT, LE, GT or GE
- **value** – specifies the value to be used in the conditional test.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.6.1.2 Elements

- **<Actions>** (required) specifies a list of Actions to be executed. Any valid Action can be contained in the list of Actions.

- **<Else>** (optional) element can optional specify a list of Actions to be executed. Any valid Action can be contained in the list of Actions

  Attribute of <Else>:

  - **exitrc** (optional) – specifies an exit return code for the else part of the conditional test
  - **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

  Elements of <Else>:

  - **<Actions>** (required) specifies a list of Actions to be executed. Any valid Action can be contained in the list of Actions.

4.6.1.3 Field Names that can be Tested

The following table shows the field names of each action that can be tested. Note: ***Field names are case sensitive.***

Action Name	Field name	Field Type	Description
Append	createDir	Boolean	Create the target directory if it does not exist.
	fileCount	Integer	The number of files processed.
	srcFilename	String	The source file name.
	toDir	String	The full path to the target directory.
	toFile	String	The target file name.
Convert	createDir	Boolean	Create the target directory if it does not exist.
	fileCount	Integer	The number of files processed.
	mode	Integer	The conversion to take place.
	srcFilename	String	The source file name.
	toDir	String	The full path to the target directory.
	toFile	String	The target file name.
Copy	createDir	Boolean	Create the target directory if it does not exist.
	fileCount	Integer	The number of files processed.
	srcFilename	String	The source file name.
	toDir	String	The full path to the target directory.
	toFile	String	The target file name.
DecryptFile	fileCount	Integer	The number of files processed.
	srcFilename	String	The source file name.
	toFilename	String	The target file name.
Echo	Text	String	String to be outputted.
EncryptFile	fileCount	Integer	The number of files processed.
	srcFilename	String	The source file name.
	toFilename	String	The target file name.
Execute	xmlFile	String	The name of the Job XML file.
Ftp	getFileCount	Integer	Number of files retrieved
	petFileCount	Integer	Number of files stored
	xmlFile	String	The name of the FTP XML file.
HttpGetFile	createDir	Boolean	Create the target directory if it does not exist.
	fileCount	Integer	The number of files processed.
	toDir	String	The full path to the target directory.

Action Name	Field name	Field Type	Description
	toFile	String	The target file name.
	url	String	URL of the file on the HTTP server.
HttpPutFile	fileCount	Integer	The number of files processed.
	FormDataName	Integer	Name of the form-data-name on HTTP server.
	srcFilename	String	The source file name.
	url	String	URL of the file on the HTTP server.
Launch	xmlFile	String	The name of the UFM Workflow XML file.
Loop	by	Integer	Increment value for the loop
	end	Integer	Ending value for the loop
	start	Integer	Starting value for the loop
MakeDir	fileCount	Integer	The number of files processed.
	dir	String	The full path to the target directory.
Merge	createDir	Boolean	Create the target directory if it does not exist.
	fileCount	Integer	The number of files processed.
	srcFilenames	String	The names of the source files.
	toDir	String	The full path to the target directory.
	toFile	String	The target file name.
MergeSort	column	Integer	The column number to begin the sort
	createDir	Boolean	Create the target directory if it does not exist.
	fileCount	Integer	The number of files processed.
	order	Character	The sort order (A or D)
	srcFilenames	String	The names of the source files.
	toDir	String	The full path to the target directory.
	toFile	String	The target file name.
Move	createDir	Boolean	Create the target directory if it does not exist.
	fileCount	Integer	The number of files processed.
	srcFilename	String	The source file name.
	toFile	String	The target file name.
	toDir	String	The full path to the target directory.
MQPutQuit	CCDTFile	String	The path and name of the CCDT file.
	fileCount	Integer	The number of files processed.
	msgCount	Integer	The number of messages processed.

Action Name	Field name	Field Type	Description
	mqXMLFile	String	The name of the MQ XML file.
	qMgrName	String	The name of the queue manager.
	qName	String	The name of the queue
MQReceive	append	Boolean	Append to data to an existing file
	backOutQName	String	The name of the backout queue
	createDir	Boolean	Create the target directory if it does not exist.
	CCDTFile	String	The path and name of the CCDT file.
	defaultDirectory	String	The default directory to place the files into
	defaultFilename	String	The default file name for the incoming file
	defaultJobXmlFile	String	The default name of the Job XML file
	fileCount	Integer	The number of files processed.
	msgCount	Integer	The number of messages processed.
	mqXMLFile	String	The name of the MQ XML file.
	overrideDirectoryName	Boolean	Override the incoming directory name?
	overrideFileName	Boolean	Override the incoming file name?
	overrideJobName	Boolean	Override the incoming Job XML file name?
	processType	Boolean	Type of process that the Receive action is running [D/T/E/S]
	qMgrName	String	The name of the queue manager.
	qName	String	The name of the queue
MQSend	archiveDirectory	String	The name of the archive directory
	archiveFile	Boolean	A flag to signal if the file should be archived
	archiveFileName	String	The name to be given for the archive file
	backOutQName	String	The name of the backout queue
	CCDTFile	String	The path and name of the CCDT file.
	deleteFile	Boolean	Should the file be deleted after processing
	fileCount	Integer	The number of files processed.
	format	Character	Message format
	jobXmlFile	String	The name of the Job XML file
	msgCount	Integer	The number of messages processed.
	mqXMLFile	String	The name of the MQ XML file.
	overrideDirectoryName	Boolean	Override the incoming directory name?
	overrideFileName	Boolean	Override the incoming file name?

Action Name	Field name	Field Type	Description
	overrideJobName	Boolean	Override the incoming Job XML file name?
	persistence	Boolean	Is the message persistent or not
	priority	Integer	Priority value for the message [0-9]
	remoteDirectory	String	The name of the directory on the remote server
	remoteFilename	String	The name of the file to be used on the remote server
	remoteJobXmlFile	String	The Job XML file to be executed on the remote server
	qMgrName	String	The name of the queue manager.
	qName	String	The name of the queue
	srcFilename	String	The source file name.
RemoveDir	dir	String	The full path to the target directory.
	fileCount	Integer	The number of files processed.
	removeAll	Boolean	Y/N - Remove all directories and/or files in the target directory
Rename	fileCount	Integer	The number of files processed.
	srcFilename	String	The source file name.
	toFile	String	The target file name.
ReplaceText	column	Integer	The column number to begin the search and replace
	createDir	Boolean	Create the target directory if it does not exist.
	fileCount	Integer	The number of files processed.
	findText	String	The text for the search
	replaceText	String	The replacement text
	row	Integer	The row number to begin the search and replace
	srcFilename	String	The names of the source file.
	toDir	String	The full path to the target directory.
	toFile	String	The target file name.
Schedule	xmlFile	String	The name of the UFM Workflow XML file.
Scp	getFileCount	Integer	Number of files retrieved
	petFileCount	Integer	Number of files stored
	xmlFile	String	The name of the SCP XML file.
SendEmail	msgCount	Integer	The number of messages processed.

Action Name	Field name	Field Type	Description
	xmlFile	String	The name of the UFM Workflow XML file.
SFtp	getFileCount	Integer	Number of files retrieved
	petFileCount	Integer	Number of files stored
	xmlFile	String	The name of the SFTP XML file.
Sleep	hours	Integer	The number of hours to sleep for
	minutes	Integer	The number of minutes to sleep for
	seconds	Integer	The number of seconds to sleep for
	milliseconds	Integer	The number of milliseconds to sleep for
Sort	column	Integer	The column number to begin the sort
	createDir	Boolean	Create the target directory if it does not exist.
	fileCount	Integer	The number of files processed.
	order	Character	The sort order (A or D)
	srcFilename	String	The source file name.
	toDir	String	The full path to the target directory.
	toFile	String	The target file name.
SortUnique	column	Integer	The column number to begin the sort
	createDir	Boolean	Create the target directory if it does not exist.
	fileCount	Integer	The number of files processed.
	order	Character	The sort order (A or D)
	srcFilename	String	The source file name.
	toDir	String	The full path to the target directory.
	toFile	String	The target file name.
Tar	createDir	Boolean	Create the target directory if it does not exist.
	fileCount	Integer	The number of files processed.
	srcFilename	String	The source file name.
	toDir	String	The full path to the target directory.
	toFile	String	The target tar file name.
Touch	createDir	Boolean	Create the target directory if it does not exist.
	filenames	String	The file names to be touched
UnTar	createDir	Boolean	Create the target directory if it does not exist.
	fileCount	Integer	The number of files processed.
	tarFilename	String	The source tar file name.
	toDir	String	The full path to the target directory.

Action Name	Field name	Field Type	Description
UnZip	createDir	Boolean	Create the target directory if it does not exist.
	fileCount	Integer	The number of files processed.
	zipFilename	String	The source zip file name.
	toDir	String	The full path to the target directory.
Watch	pollInterval	Integer	In seconds, how often to check for file(s)
WriteText	append	Boolean	Append the data to the target file
	createDir	Boolean	Create the target directory if it does not exist.
	fileCount	Integer	The number of files processed.
	text	String	The text to be written to a file
	toDir	String	The full path to the target directory.
	toFile		The name of the target file
Zip	createDir	Boolean	Create the target directory if it does not exist.
	fileCount	Integer	The number of files processed.
	srcFilename	String	The source file name.
	toDir	String	The full path to the target directory.
	toFile	String	The target zip file name.

4.6.1.4 Example

An example of an If Action:

```
<If actionname="Receive" fieldname="msgCount" operator="GT" value="0">
  <Actions>
     <Touch>
       <File dir="C:\temp">test.xml</File>
     </Touch>
  </Actions>
</If>
```

An example of an If/Else Action:

```
<If actionname="Receive" fieldname="msgCount" operator="GT" value="0">
  <Actions>
    <Touch>
      <File dir="C:\temp">test.xml</File>
    </Touch>
  </Actions>
<Else>
  <Actions>
    <Copy todir="C:\temp\ufm" tofile="${FILE}_${DATE}">
      <File dir="data">test.xml</File>
    </Copy>
  </Actions>
</Else>
</If>
```

An example of an If/Else Action with a non-zero exit return code that causes an abnormal exit to :

```
<If actionname="Receive" fieldname="msgCount" operator="GT" value="0">
  <Actions>
    <Touch>
      <File dir="C:\temp">test.xml</File>
    </Touch>
  <Actions>
<Else exitrc=8 />
</If>
```

An example of an If/Else Action containing another If action:

```
<If actionname="Receive" fieldname="msgCount" operator="GT" value="0">
  <Actions>
    <Touch>
      <File dir="C:\temp">test.xml</File>
    </Touch>
    <If actionname="Touch" fieldname="fileCount" operator="GT" value="0">
      <Copy todir="C:\temp\ufm" tofile="${FILE}_${DATE}">
        <File dir="data">test.xml</File>
      </Copy>
    </If>
  <Actions>
<Else exitrc=8 />
</If>
```

4.6.2 <Loop> Element

This section describes how to invoke the Loop Action. The Loop Action is used to iterate over a range of values.

4.6.2.1 Attributes

- **start** (required) specifies start value for the loop counter. Default value is 1.
- **end** (required) specifies end value for the loop counter. Default value is 1.
- **by** (optional) specifies increment value for the loop counter. Default value is 1.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.6.2.2 Elements

- **<Actions>** (required) specifies a list of Actions to be executed. Any valid Action can be contained in the list of Actions.

4.6.2.3 Example

An example of a Loop Action:

```
<Loop start="1" end="3" by="1">

  <Actions>

    <Echo>This is a test message # ${LOOP_COUNTER}.</Echo>

    <Send delete="N" format="S">
      <QMgrName>MQWT1</QMgrName>
      <QueueName>TEST.Q1</QueueName>
      <File>data\text_${LOOP_COUNTER}.xml</File>
    </Send>

  </Actions>

</Loop>
```

Note: The ${LOOP_COUNTER} variable is by the user as a place holder to the loop counter number from the Loop element. UFM will transform the variable when the Action is executed.

4.6.3 <Sleep> Element

This section describes how to invoke the Sleep Action. The Sleep Action is used to pause the Workflow for a period of time.

4.6.3.1 Attributes

- **hours** (optional) specifies number of hours to sleep for.
- **minutes** (optional) specifies number of minutes to sleep for.
- **seconds** (optional) specifies number of seconds to sleep for.
- **milliseconds** (optional) specifies number of milliseconds to sleep for.
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.6.3.2 Elements

None

4.6.3.3 Example

An example of a Sleep Action:

```
<Sleep hours="2" minutes="30" seconds="10" />
```

4.7 Other Action Elements

There are 4 other action elements and are detailed below.

4.7.1 <Echo> Element

This section describes how to invoke the Echo Action. The Echo Action is used to write/output a text string to the log file.

4.7.1.1 Attributes

- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.7.1.2 Elements

None.

4.7.1.3 Example

An example of an Echo Action:

```
<Echo>This is a test message.</Echo>
```

4.7.2 <Execute> Element

This section describes how to invoke the Execute Action. The Execute Action is used to run an external program.

4.7.2.1 Attributes

- **xmlfile** (required) specifies the name of the Job XML file (see section 7 for more details)
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.7.2.2 Elements

None

4.7.2.3 Example

An example of an Execute Action:

```
<Execute xmlfile="abcrun.xml" />
```

4.7.3 <Launch> Element

This section describes how to invoke the Launch Action. The Launch Action is used to invoke an UFM Workflow XML file.

4.7.3.1 Attributes

- **xmlfile** (required) specifies the name of the UFM Workflow XML file
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.7.3.2 Elements

None.

4.7.3.3 Example

An example of a Launch Action:

```
<Launch xmlfile="ufm_workflow.xml" />
```

4.7.4 <Schedule> Element

This section describes how to invoke the Schedule Action. The Schedules Action is used to invoke an UFM Workflow XML file at a specific date and/or time.

4.7.4.1 Attributes

- **xmlfile** (required) specifies the name of the Schedule XML file (see section 9 for more details)
- **label** – specifies a label name for the action. The default value is 'default'.
- **onerrorfail** – [Y / N] – specifies whether or not the Workflow should abnormally terminate if the Action fails. The default value is 'Y'.

4.7.4.2 Elements

None.

4.7.4.3 Example

An example of an Schedule Action:

```
<Schedule xmlfile="myschedule.xml" />
```

5 UFM_Email_SMTP XML File

This section describes how to create an Email SMTP XML file for use by UFM SendEmail Action. The Email SMTP XML file contains information to send an email message via SMTP.

5.1 <UFM_Email_SMTP> Root Element

<UFM_Email_SMTP> is the root element for the UFM_Email_SMTP XML file. The UFM_Email_SMTP root element is comprised of 1 element (Email).

Note: The UFM_Email_SMTP XML files must be stored in the <UFM_Install_PATH>\email\ directory. i.e. C:\Capitalware\UFM\email\

5.1.1 <Hostname> Element

The Hostname element contains the SMTP hostname or IP address.

Attributes:

- "**authentication**" (optional) specifies if the connection will use UserID and Password authentication. If so, then the UserID and Password elements must be specified.
- "**ssl**" (optional) specifies if the connection will use SSL

e.g.
<Hostname ssl="Y" authentication="Y">mail.acme.com</Hostname>

5.1.2 <Port> Element

The Port element contains the SMTP port number. The default value is 25. Note: Certain SMTP servers require that the user use either 465 or 587 as the port number.

5.1.3 <UserID> Element

The UserID element contains the UserID to be used when authenticating against an SMTP server.

5.1.4 <Password> Element

The Password element contains the password to be used when authenticating against an SMTP server.

5.1.5 <FromEmailAddress> Element

The FromEmailAddress element contains the “From” email address.

5.1.6 <ToEmailAddress> Element

The ToEmailAddress element contains the “To” email addresses of the email to be sent. Separate each email address with a comma (',').
e.g.
<ToEmailAddress>fred@acme.com,barney@acme.com,pebbles@acme.com</ToEmailAddress>

5.1.7 <Subject> Element

The Subject element contains the subject of the email to be sent.

5.1.8 <MessageText> Element

The MessageText element contains the message text (body) of the email to be sent.

5.2 Sample

To create an UFM_Email_SMTP XML file, open a text editor and input one or more of the above elements as shown in the example below:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_Email_SMTP SYSTEM "UFM_Email_SMTP.dtd">
<UFM_Email_SMTP>
  <Hostname ssl="Y" authentication="Y">mail.acme.com</Hostname>
  <Port>465</Port>
  <UserID>mrslate@acme.com</UserID>
  <Password>mypwd</Password>
  <FromEmailAddress>mrslate@acme.com</FromEmailAddress>
  <ToEmailAddress>fred@acme.com,barney@acme.com</ToEmailAddress>
  <Subject>test subject</Subject>
  <MessageText>This is a test message
The second line of the test message
The third line of the test message</MessageText>
</UFM_Email_SMTP>
```

Note: The UFM_Email_SMTP XML files must be stored in the <UFM_Install_PATH>\email\ directory. i.e. C:\Capitalware\UFM\email\

6 UFM_Ftp XML File

This section describes how to create an Ftp XML file for use by UFM Ftp Action. The Ftp XML file contains information to process FTP commands with a remote FTP server.

6.1 <UFM_Ftp> Root Element

<UFM_Ftp> is the root element for the UFM_Ftp XML file. The UFM_Ftp root element is comprised of 4 elements: Hostname, UserID, Password and FtpCmds

Note: The UFM_Ftp XML files must be stored in the <UFM_Install_PATH>\ftp\ directory. i.e. C:\Capitalware\UFM\ftp\

6.1.1 <Hostname> Element

The Hostname element contains the ftp hostname or IP address.

6.1.2 <UserID> Element

The UserID element contains the UserID to be used when authenticating against an ftp server.

6.1.3 <Password> Element

The Password element contains the password to be used when authenticating against an ftp server.

6.1.4 <FtpCmds> Element

The FtpCmds element contains the following elements: Put, Get, Type, Cd, MkDir, RMDir and Delete.

6.1.4.1 <Cd> Element

The Cd element specifies that directory on the remote ftp server is to be changed.

Attributes:

- "**remotedir**" specifies the directory on the remote ftp server

6.1.4.2 <Delete> Element

The Delete element specifies the remote file to be deleted.

Attributes:

- "**remotedir**" specifies the directory on the remote ftp server
- "**remotefile**" specifies the name of remote file

6.1.4.3 <Get> Element

The Get element specifies that a remote file is to be retrieved from a remote ftp server.

Attributes:

- "**remotedir**" specifies the directory on the remote ftp server
- "**remotefile**" specifies the name of remote file

Elements:

- **<File>** (required) specifies the name of the local file
  Attributes of <File>:
  - "dir" (optional) specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

6.1.4.4 <MkDir> Element

The MkDir element specifies that a directory is to be created on the remote ftp server.

Attributes:

- "**remotedir**" specifies the directory on the remote ftp server

6.1.4.5 <Put> Element

The Put element specifies that a local file (or many files) is to be uploaded to a remote ftp server.

Attributes:

- "**remotedir**" specifies the directory on the remote ftp server
- "**remotefile**" specifies the name of remote file

Elements:

- **<File>** (required) specifies the name of the local file
  Attributes of <File>:
  - "dir" (optional) specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

6.1.4.6 <RmDir> Element

The RmDir element specifies that a remote directory is to be deleted on the remote ftp server.

Attributes:

- "**remotedir**" specifies the directory on the remote ftp server

6.1.4.7 <Type> Element

The Type element specifies that the transfer type is to be changed.

Attributes:

- "**mode**" [I / A] specifies the transfer type to be used – "I" is for Binary and "A" is for ASCII.

6.2 Sample

To create an UFM_Ftp XML file, open a text editor and input one or more of the above elements as shown in the example below:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_Ftp SYSTEM "UFM_Ftp.dtd">
<UFM_Ftp>
  <Hostname>ftp.acme.com</Hostname>
  <UserID>mrslate@acme.com</UserID>
  <Password>mypwd</Password>

  <FtpCmds>
    <Type mode="A" />

    <Put remotedir="newtest">
      <File dir="C:\temp\UFM\ftp\test">*.txt</File>
    </Put>

  </FtpCmds>
</UFM_Ftp>
```

Note: The UFM_Ftp XML files must be stored in the <UFM_Install_PATH>\ftp\ directory. i.e. C:\Capitalware\UFM\ftp\

7 UFM_Job XML File

This section describes how to create a Job XML file for use by UFM Execute Action. The Job XML file contains configuration information to run an external program.

7.1 <UFM_Job> Root Element

<UFM_Job> is the root element for the UFM_Job XML file. The UFM_Job root element is comprised of 1 element (Job).

If the user is using a Send Action to send a single file, and invoking a local external command, do not use a Command's ***wait*** attribute with a value of "N", since the main process may terminate before the child completes running the external program.

Note: The UFM_Job XML files must be stored in the <UFM_Install_PATH>\jobs\ directory.
i.e. C:\Capitalware\UFM\jobs\

7.2 <Job> Element

The Job element is comprised of a Command element, an optional Directory element and one or more Parm elements.

7.2.1 <Command> Element

The Command element is a required element that specifies the external program to be executed.

Attributes:

- "**wait**" [Y / N] specifies the directory in which the file is located
  e.g. <Command wait="Y">C:\temp\abc.exe</Command>

7.2.2 <Directory> Element

The Directory element is optional and if used specifies the working directory for the command.

7.2.3 <Parm> Element

The Parm element is optional and if used specifies the parameter for the command. Note: Place each parameter in a separate <Parm> element.

7.3 Sample

To create an UFM_Job XML file, open a text editor and input one or more of the above elements as shown in the example below:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_Job SYSTEM "UFM_Job.dtd">
<UFM_job>
  <Job>
    <Command wait='N'>C:\test\abc.exe</Command>
    <Directory>C:\test</Directory>
    <Parm>-f</Parm>
    <Parm>data/test.xml</Parm>
    <Parm>-m</Parm>
    <Parm>MQWT1</Parm>
    <Parm>-q</Parm>
    <Parm>TEST.Q10</Parm>
    <Parm>-d</Parm>
    <Parm>C:\Temp</Parm>
  </Job>
</UFM_job>
```

Note: The UFM_Job XML files must be stored in the <UFM_Install_PATH>\jobs\ directory.
i.e. C:\Capitalware\UFM\jobs\

8 UFM_MQ XML File

This section describes how to create an UFM_MQ XML file for use by UFM. The UFM_MQ XML file contains the MQ configuration information for connecting to a remote queue manager as described below.

8.1 <UFM_MQ> Root Element

<UFM_MQ> is the root element for the UFM_MQ XML file. The UFM_MQ root element is comprised of 19 elements. According to the user's needs, one or more of the elements can be used.

8.1.1 <QMgrName> Element

The QMgrName element (optional) contains the name of the queue manager.

8.1.2 <QueueName> Element

The QueueName element (optional) contains the name of the queue.

Attributes:

- "**qmgrname**" (optional) specifies the name of a remote queue manager
  e.g. <QueueName qmgrname="MQC1">TEST.Q1</QueueName>

8.1.3 <DistributionList> Element

The DistributionList element (optional) specifies a list of the queue names. The DistributionList element contains more than one QueueName element.

8.1.3.1 <QueueName> Element

The QueueName element (optional) contains the name of the queue.

Attributes:

- "**qmgrname**" (optional) specifies the name of a remote queue manager
  e.g. <QueueName qmgrname="MQC1">TEST.Q1</QueueName>

8.1.4 <CCDTFile> Element

The CCDTFile element (optional) contains a CCDT file.

8.1.5 <Hostname> Element

The Hostname element (optional) contains the hostname or IP address.

8.1.6 <Port> Element

The Port element (optional) contains the port number.

8.1.7 <ChannelName> Element

The ChannelName element (optional) contains the name of the channel.

8.1.8 <UserID> Element

The UserID element (optional) contains the UserID to be used for the connection.

8.1.9 <Password> Element

The Password element (optional) contains the Password to be used for the connection.

8.1.10 <SecurityExit> Element

The SecurityExit element (optional) contains the name of the security exit.

8.1.11 <SecurityExitPath> Element

The SecurityExitPath element (optional) specifies the path to the security exit.

8.1.12 <SecurityExitData> Element

The SecurityExitData element (optional) specifies the data for the security exit.

8.1.13 <CipherSuiteName> Element

The CipherSuiteName element (optional) contains the name of the Cipher Suite.

8.1.14 <DistinguishedName> Element

The DistinguishedName element (optional) contains the name of the Distinguished.

8.1.15 <TrustedStore> Element

The TrustedStore element (optional) contains the name of the Trusted Store.

8.1.16 <TrustedStorePasswd> Element

The TrustedStorePassword element (optional) contains the password of the Trusted Store.

8.1.17 <KeyStore> Element

The KeyStore element (optional) contains the name of the Key Store.

8.1.18 <KeyStorePasswd> Element

The KeyStorePassword element (optional) contains the password name of the Key Store.

8.1.19 <LDAPServer> Element

The LDAPServer element (optional) contains the name of LDAP server hostname or IP address.

8.1.20 <LDAPServerPort> Element

The LDAPServerPort element (optional) contains the LDAP server port number.

8.2 Sample

To create an UFM_MQ XML file, open a text editor and input one or more of the above elements into the UFM_MQ XML file.

An example of a standard UFM_MQ XML file for connecting to a local queue manager and specifying a single queue:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_MQ SYSTEM "UFM_MQ.dtd">
<UFM_MQ>
    <QMgrName>MQWT1</QMgrName>
    <QueueName>TEST.Q1</QueueName>
    <UserID>abcuid</UserID>
</UFM_MQ>
```

An example of a standard UFM_MQ XML file for connecting to a remote queue manager and specifying a single queue:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_MQ SYSTEM "UFM_MQ.dtd">
<UFM_MQ>
    <QMgrName>MQWT1</QMgrName>
    <QueueName>TEST.Q1</QueueName>
    <Hostname>10.10.10.10</Hostname>
    <ChannelName>SYSTEM.DEF.SVRCONN</ChannelName>
    <Port>1414</Port>
    <UserID>abcuid</UserID>
</UFM_MQ>
```

An example of a UFM_MQ XML file for connecting to a remote queue manager and specifying a list of queues that each message will be sent to:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_MQ SYSTEM "UFM_MQ.dtd">
<UFM_MQ>
    <QMgrName>MQWT1</QMgrName>
    <DistributionList>
      <QueueName>TEST.Q1</QueueName>
      <QueueName>TEST.Q2</QueueName>
      <QueueName>TEST.Q3</QueueName>
    </DistributionList>
    <Hostname>10.10.10.10</Hostname>
    <ChannelName>SYSTEM.DEF.SVRCONN</ChannelName>
    <Port>1414</Port>
    <UserID>abcuid</UserID>
</UFM_MQ>
```

Note: The UFM_MQ XML files must be stored in the <UFM_Install_PATH>\mq\ directory. i.e. C:\Capitalware\UFM\mq\

9 UFM_Schedule XML File

This section describes how to create a Schedule XML file for use by UFM Schedule Action. The Schedule XML file contains information for starting an UFM Workflow XML at a particular date and/or time.

9.1 <UFM_Schedule> Root Element

<UFM_Schedule> is the root element for the UFM_Schedule XML file.

Note: The UFM_Schedule XML files must be stored in the <UFM_Install_PATH>\schedule\ directory. i.e. C:\Capitalware\UFM\schedule\

9.1.1 <HolidayFile> Element

The HolidayFile element (optional) specifies a UFM_Holiday XML file. The Holiday XML file is used as an exclusion date file.

9.1.2 <Schedule> Element

The Schedule element (required) contains the information for starting an UFM Workflow XML at a particular date and/or time.

9.1.2.1 Attributes

- **xmlfile** (required) specifies the name of the UFM Workflow XML file

9.1.2.2 Elements

- **<Minute>** (optional) specifies the minute(s) for the scheduled event. The default value is "*" which means it will be run once every minute. Valid values are 0-59. The user can input a single minute value (i.e. 0) or a group of minutes (i.e. 0,10,20,30,40,50).

- **<Hour>** (optional) specifies the hour(s) for the scheduled event. The default value is "*". Valid values are 0-23. The user can input a single hour value (i.e. 0) or a group of hours (i.e. 0,6,12,18).

- **<DayOfMonth>** (optional) specifies the DayOfMonth for the scheduled event. The default value is "*". Valid values are 1-31. The user can input a single day value (i.e. 1) or a group of hours (i.e. 0,6,12,18).

- **<Month>** (optional) specifies the month(s) for the scheduled event. The default value is "*". Valid values are 1-12 (or use the names of months). The user can input a single month value (i.e. 1) or a group of hours (i.e. 1,4,7,10).

- **<DayOfWeek>** (optional) specifies the day(s) for the scheduled event. The default value is "*". Valid values are 0-7 (0 or 7 for Sunday or use the names of days). The

user can input a single day value (i.e. 2 or TUESDAY) or a group of hours (i.e. 1,3,5 or MONDAY, WEDNESDAY,FRIDAY).

- **<Year>** (optional) specifies the year(s) for the scheduled event. The default value is "*". The user can input a single year value (i.e. 2011) or a group of years (i.e. 2011,2013,2015).

9.2 <UFM_Holiday> Root Element

<UFM_Holiday> is the root element for the UFM_Holiday XML file. The Holiday XML file is used as an exclusion date file. In other words, if a scheduled event is to be launched but the current date matches a holiday from the Holiday XML file then the scheduled event is not launched.

Note: The UFM_Holiday XML files must be stored in the <UFM_Install_PATH>\schedule\ directory. i.e. C:\Capitalware\UFM\schedule\

9.2.1 <Holiday> Element

The Holiday element (required) contains the information for starting an UFM Workflow XML at a particular date and/or time.

9.2.1.1 Attributes

None.

9.2.1.2 Elements

- **<Day>** (optional) specifies the Day for the scheduled event. The default value is "*". Valid values are 1-31.

- **<Month>** (optional) specifies the month for the scheduled event. The default value is "*".

- **<Year>** (optional) specifies the year for the scheduled event. The default value is "*".

9.3 Sample

To create an **UFM_Schedule** XML file, open a text editor and input one or more of the above elements as shown in the example below (this examples launches an UFM Workflow every minute):

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_Schedule SYSTEM "UFM_Schedule.dtd">
<UFM_Schedule>

  <HolidayFile>holiday.xml</HolidayFile>

  <Schedule xmlfile="ufm_workflow.xml">
    <Minute>*</Minute>
    <Hour>*</Hour>
    <DayOfMonth>*</DayOfMonth>
    <Month>*</Month>
    <DayOfWeek>*</DayOfWeek>
    <Year>*</Year>
  </Schedule>

</UFM_Schedule>
```

Note: The UFM_Schedule XML files must be stored in the <UFM_Install_PATH>\schedule\ directory. i.e. C:\Capitalware\UFM\schedule\

To create an **UFM_Holiday** XML file, open a text editor and input one or more of the above elements as shown in the example below:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_Holiday SYSTEM "UFM_Holiday.dtd">
<UFM_Holiday>

  <Holiday name="New Year's Day">
    <Month>1</Month>
    <Day>1</Day>
  </Holiday>

  <Holiday name="July 4th">
    <Month>7</Month>
    <Day>4</Day>
  </Holiday>

  <Holiday name="Christmas">
    <Month>25</Month>
    <Day>12</Day>
  </Holiday>

</UFM_Holiday>
```

Note: The UFM_Holiday XML files must be stored in the <UFM_Install_PATH>\schedule\ directory. i.e. C:\Capitalware\UFM\schedule\

10 UFM_Scp XML File

This section describes how to create an Scp XML file for use by UFM Scp Action. The Scp XML file contains information to process SCP commands with a remote server.

10.1 <UFM_Scp> Root Element

<UFM_Scp> is the root element for the UFM_Scp XML file. The UFM_Scp root element is comprised of 4 elements: Hostname, UserID, Password and ScpCmds.

Note: The UFM_Scp XML files must be stored in the <UFM_Install_PATH>\scp\ directory. i.e. C:\Capitalware\UFM\scp\

10.1.1 <Hostname> Element

The Hostname element contains the hostname or IP address.

10.1.2 <UserID> Element

The UserID element contains the UserID to be used when authenticating against an server.

10.1.3 <Password> Element

The Password element contains the password to be used when authenticating against an server.

10.1.4 <ScpCmds> Element

The ScpCmds element contains the following elements: CopyTo and CopyFrom.

10.1.4.1 <CopyFrom> Element

The CopyFrom element specifies that a remote file is to be retrieved from a remote server.

Attributes:

- "**remotedir**" specifies the directory on the remote sftp server
- "**remotefile**" specifies the name of remote file

Elements:

- **<File>** (required) specifies the name of the local file
  Attributes of <File>:
  - "dir" (optional) specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

10.1.4.2 <CopyTo> Element

The CopyTo element specifies that a local file (or many files) is to be uploaded to a remote server.

Attributes:

- "**remotedir**" specifies the directory on the remote sftp server
- "**remotefile**" specifies the name of remote file

Elements:

- **<File>** (required) specifies the name of the local file
  Attributes of <File>:
  - "dir" (optional) specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

10.2 Sample

To create an UFM_Scp XML file, open a text editor and input one or more of the above elements as shown in the example below:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_Scp SYSTEM "UFM_Scp.dtd">
<UFM_Scp>
  <Hostname>scp.acme.com</Hostname>
  <UserID>mrslate@acme.com</UserID>
  <Password>mypwd</Password>

  <ScpCmds>

    <CopyTo remotedir="newtest">
      <File dir="C:\temp\UFM\scp\test">text.txt</File>
    </CopyTo>

  </ScpCmds>
</UFM_Scp>
```

Note: The UFM_Scp XML files must be stored in the <UFM_Install_PATH>\scp\ directory. i.e. C:\Capitalware\UFM\scp\

11 UFM_SFtp XML File

This section describes how to create an SFtp XML file for use by UFM SFtp Action. The SFtp XML file contains information to process SFTP commands with a remote SFTP server.

11.1 <UFM_SFtp> Root Element

<UFM_SFtp> is the root element for the UFM_SFtp XML file. The UFM_SFtp root element is comprised of 4 elements: Hostname, UserID, Password and SFtpCmds

Note: The UFM_SFtp XML files must be stored in the <UFM_Install_PATH>\sftp\ directory. i.e. C:\Capitalware\UFM\sftp\

11.1.1 <Hostname> Element

The Hostname element contains the sftp hostname or IP address.

11.1.2 <UserID> Element

The UserID element contains the UserID to be used when authenticating against an sftp server.

11.1.3 <Password> Element

The Password element contains the password to be used when authenticating against an sftp server.

11.1.4 <SFtpCmds> Element

The SFtpCmds element contains the following elements: Put, Get, Type, Cd, MkDir, RMDir and Delete.

11.1.4.1 <Cd> Element

The Cd element specifies that directory on the remote sftp server is to be changed.

Attributes:

- "**remotedir**" specifies the directory on the remote sftp server

11.1.4.2 <Delete> Element

The Delete element specifies the remote file to be deleted.

Attributes:

- "**remotedir**" specifies the directory on the remote sftp server
- "**remotefile**" specifies the name of remote file

11.1.4.3 <Get> Element

The Get element specifies that a remote file is to be retrieved from a remote sftp server.

Attributes:

- "**remotedir**" specifies the directory on the remote sftp server
- "**remotefile**" specifies the name of remote file

Elements:

- **<File>** (required) specifies the name of the local file
  Attributes of <File>:
  - "dir" (optional) specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

11.1.4.4 <MkDir> Element

The MkDir element specifies that a directory is to be created on the remote sftp server.

Attributes:

- "**remotedir**" specifies the directory on the remote sftp server

11.1.4.5 <Put> Element

The Put element specifies that a local file (or many files) is to be uploaded to a remote sftp server.

Attributes:

- "**remotedir**" specifies the directory on the remote sftp server
- "**remotefile**" specifies the name of remote file

Elements:

- **<File>** (required) specifies the name of the local file
  Attributes of <File>:
  - "dir" (optional) specifies the directory where the file is located

  e.g. <File dir="C:\test">data.txt</File>

11.1.4.6 <RmDir> Element

The RmDir element specifies that a remote directory is to be deleted on the remote sftp server.

Attributes:

- "**remotedir**" specifies the directory on the remote sftp server

11.1.4.7 <Type> Element

The Type element specifies that the transfer type is to be changed.

Attributes:

- "**mode**" [I / A] specifies the transfer type to be used – "I" is for Binary and "A" is for ASCII.

11.2 Sample

To create an UFM_SFtp XML file, open a text editor and input one or more of the above elements as shown in the example below:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_SFtp SYSTEM "UFM_SFtp.dtd">
<UFM_SFtp>
  <Hostname>sftp.acme.com</Hostname>
  <UserID>mrslate@acme.com</UserID>
  <Password>mypwd</Password>

  <SFtpCmds>
    <Type mode="A" />

    <Put remotedir="newtest">
      <File dir="C:\temp\UFM\sftp\test">*.txt</File>
    </Put>

  </SFtpCmds>
</UFM_SFtp>
```

Note: The UFM_SFtp XML files must be stored in the <UFM_Install_PATH>\sftp\ directory. i.e. C:\Capitalware\UFM\sftp\

12 UFM Tokens

This section describes how token replacement will be done in UFM.

12.1 Tokens

12.1.1 Date / Time Tokens

The following is a list of the Date and Time tokens that will be replaced:

Token	Description
${DATE}	Date in the format: YYYY_MM_DD e.g. 2009_05_06
${DATETIME}	Date in the format: YYYY_MM_DD_HH_MM_SS.SSS e.g. 2009_04_30_22_10_07.123
${DATEM}	Substract the user-defined Property "${DATEMINUS}" from the current date. Date in the format: YYYY_MM_DD e.g. 2009_05_06
${DATE-1} ${DATE-2} ${DATE-3} ${DATE-4} ${DATE-5} ${DATE-6} ${DATE-7}	Substract the value (i.e. 1) from the current date. Date in the format: YYYY_MM_DD e.g. 2009_05_06
${DATEP}	Add the user-defined Property "${DATEPLUS}" to the current date. Date in the format: YYYY_MM_DD e.g. 2009_05_06
${DATE+1} ${DATE+2} ${DATE+3} ${DATE+4} ${DATE+5} ${DATE+6} ${DATE+7}	Add the value (i.e. 1) to the current date. Date in the format: YYYY_MM_DD e.g. 2009_05_06
${TIME}	Time in the format: HH_MM_SS.SSS e.g. 20_37_55.445
${YYYY}	Year in the format: YYYY e.g. 2009
${MM}	Month in the format: MM e.g. 05
${DD}	Day in the format: DD e.g. 14
${HH}	Hour in the format: HH e.g. 19
${hh}	Round the minutes to the next half hour: hh e.g.00 or 30
${MIN}	Minutes in the format: MM e.g. 47
${SS}	Seconds in the format: SS e.g. 33

12.1.2 File Tokens

The following is a list of the file tokens that will be replaced:

Token	Description
${FILE}	The complete “from” file name e.g. abc.txt
${FILE_PREFIX}	The prefix portion of the file name e.g. abc
${FILE_EXT}	The file extension of the file name e.g. txt

12.1.3 Index Token

The INDEX token is used to add an index counter to the file name:

Token	Description
${INDEX}	Index counter of the file currently being processed e.g. 0003

12.1.4 Loop Token

The LOOP_COUNTER token is used to add an index counter to the file name:

Token	Description
${LOOP_COUNTER}	Loop counter of the current loop iteration e.g. 2

12.1.5 Watch Token

The WATCH_FOUND_FILE token can be used with any action in the Actions element of the WatchItem element.

Token	Description
${WATCH_FOUND_FILE}	The name of the file found in the Watch action.

12.1.6 General Tokens

The following is a list of general tokens that can be replaced:

Token	Description
${ACTION}	The name of the Action component
${HOSTNAME}	The hostname of the server that UFM is running on
${IPADDRESS}	The IP address of the server that UFM is running on
${UFM_ERROR_TEXT}	The last UFM error message

12.2 Examples

```
<Copy todir="C:\temp\ufm" tofile="${FILE}.${DATE}">
  <File dir="data">test*</File>
</Copy>
```

```
<Move todir="C:\temp\ufm" tofile="${FILE_PREFIX}.${DATE}.${FILE_EXT}">
  <File dir="data">test*</File>
</Move>
```

```
<Send delete="N" format="N">
  <File>data\test.xml</File>
  <MQ>
    <QMgrName>MQWT1</QMgrName>
    <QueueName>TEST.Q1</QueueName>
  </MQ>
  <Remote>
    <Directory>C:\temp</Directory>
    <FileName>${FILE}.${DATE}</FileName>
    <Execute xmlfile="abcrun.xml" />
  </Remote>
</Send>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE UFM_Job SYSTEM "UFM_Job.dtd">
<UFM_Job>
  <Job name="alert_pgm">
    <Command wait='y'>C:\test\alert.exe</Command>
    <Parm>-m</Parm>
    <Parm>${HOSTNAME} - ${UFM_ERROR_TEXT}</Parm>
  </Job>
</UFM_Job>
```

13 Defining a Centralized Status Queue

This section describes how to define a centralized Status Queue.

13.1 NPMCLASS Set to High

Setting NPMCLASS to high means that when a queue manager is restarted, all non-persistent messages in a local queue will remain (not deleted). This is the recommended Status queue definition.

Use the following MQSC command to define a Status Queue with NPMCLASS set to high:

```
DEFINE QUEUE(CAPITALWARE.UFM.STATUS) +
       TYPE(QLOCAL) +
       MAXDEPTH(1000000) +
       MAXMSGL(4194304) +
       DEFPSIST(NO) +
       NPMCLASS(HIGH) +
       REPLACE
```

13.2 NPMCLASS Set to Normal

Setting NPMCLASS to normal means that when a queue manager is restarted, all non-persistent messages in a local queue will be removed.

Use the following MQSC command to define a Status Queue with NPMCLASS set to normal:

```
DEFINE QUEUE(CAPITALWARE.UFM.STATUS) +
       TYPE(QLOCAL) +
       MAXDEPTH(1000000) +
       MAXMSGL(4194304) +
       DEFPSIST(NO) +
       NPMCLASS(NORMAL) +
       REPLACE
```

14 Appendix A – Custom Logging

UFM supports user-defined custom logger via Apache's log4j framework.

Apache log4j has an abstract class, org.apache.log4j.AppenderSkeleton, that implements most of the functionality needed for writing a custom Appender. When AppenderSkeleton class is extended, 3 methods need to be implemented: *append*, *requiresLayout* and *close*.

A sample biz.capitalware.ufm.logging.CustomAppend is included in the source as a working sample:

```
package biz.capitalware.ufm.logging;

import org.apache.log4j.AppenderSkeleton;
import org.apache.log4j.spi.LoggingEvent;

/**
 * Sample custom appender for UFM.
 * @author Roger Lacroix, Capitalware Inc.
 * @version 1.0
 * @license Apache 2 License
 */
public class CustomAppender extends AppenderSkeleton
{
   protected void append(LoggingEvent loggingEvent)
   {
      System.out.println("CustomAppender: "+loggingEvent.getMessage());
   }

   public boolean requiresLayout()
   {
      return false;
   }

   public void close()
   {
   }
}
```

- ***void append(LoggingEvent)***: This method will be called by log4j at runtime when an output has to occur. We can extract the information from the LoggingEvent class and perform some appropriate function. Later on, we will see how to extract information from the LoggingEvent class and send a Yahoo instant message.
- ***boolean requiresLayout()***: This method is used to indicate whether the custom appender requires a Layout. Since we are going to keep things simple, we will simply return false.
- ***void close()***: This method is called by log4j runtime as a signal to release any resources such as file handles, network resources, etc. With no resources to release, we can simply keep it empty.

In addition to the above methods, one needs to write a get/set method pair for each custom property that will be configured for the appender. The setting of the properties will be done transparently at runtime by log4j.

Next, compile and your CustomAppender to the CLASSPATH of batch file or Unix shell script. If you are compiling the entire UFM source code then simply replace UFM's CustomAppender with your CustomAppender.

Finally, to use CustomAppender, the log4j.properties file will need to be updated:

```
#
# UFM
#
log4j.category.UFM=INFO, ufm, stdout, custom
#
# "ufm" appender writes to a file
#
log4j.appender.ufm=org.apache.log4j.RollingFileAppender
log4j.appender.ufm.File=log/UFM.log
log4j.appender.ufm.MaxFileSize=1000KB
log4j.appender.ufm.MaxBackupIndex=9
log4j.appender.ufm.layout=org.apache.log4j.PatternLayout
log4j.appender.ufm.layout.ConversionPattern=%d{yyyy-MM-dd HH:mm:ss} %-5p -
%m%n
#
# stdout
#
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%m%n
#
# CustomAppender
#
log4j.appender.custom=biz.capitalware.ufm.logging.CustomAppender
log4j.appender.custom.layout=org.apache.log4j.PatternLayout
log4j.appender.custom.layout.ConversionPattern=%m%n
```

15 Appendix B – Universal File Mover Upgrade Procedures

To upgrade an existing installation of Universal File Mover, please do the following in the appropriate section below.

15.1 Windows Upgrade

- Stop all instances of Watch and Receive
- Backup all UFM data files in the UFM install directory
- Delete the UFM install directory
- Unzip ufm.zip archive
- Restore the UFM data files if necessary

15.2 Unix and Linux Upgrade

- Stop all instances of Watch and Receive
- Backup all UFM data files in the UFM install directory
- Delete the UFM install directory
- Unzip ufm.zip archive
- Restore the UFM data files if necessary

15.3 IBM i Upgrade

- Stop all instances of Watch and Receive
- Backup all UFM data files in the UFM install directory
- Delete the UFM install directory
- Unzip ufm.zip archive
- Restore the UFM data files if necessary

16 Appendix C – Support

The support for Universal File Mover can be found at the following location (requires a support contract):

Online Help Desk Ticketing System at
www.capitalware.com/phpst/

By email at:
support@capitalware.com

By regular mail at:

Capitalware Inc.
Attn: Universal File Mover Support
Unit 11, 1673 Richmond Street, PMB524
London, Ontario N6G2N3
Canada

17 Appendix D – Summary of Changes

- Universal File Mover v2.0.0
  - Added RemoveItems action - it removes items in a file from a source file.
  - For Global onerrorfail="Y", if the workflow with a Watch Action encounters an error with the delete on MQSend Action then the entire workflow will stop and the OnError will run.
  - For MQReceive Action, perform tokenizing of MQ_RECEIVE_FILE before calling 'Default Execute' (if used)
  - Renamed package from "biz.capitalware" to "com.capitalware"
  - Added support for the single combined MQ JAR file called: "com.ibm.mq.allclient.jar" (i.e. IBM MQ v8 & higher).
  - Fixed a bug with the installer putting scp1.xml and sftp1.xml in the wrong directory.
  - Updated to the latest release of Jsch.
  - There are now 2 installers for Windows: 32-bit and 64-bit installers

- Universal File Mover v1.2.2
  - Fixed an issue with Zip Action not honoring the createdir attribute
  - Fixed the reported filesize value for MQSend action
  - Added try/catch for MQ version check because not everyone uses UFM with MQ

- Universal File Mover v1.2.1
  - Added reconnection code to MQReceive (and for Status Queue connection) and now it will wait forever to reconnect.
  - Added code to get and display MQ JAR version
  - Added "catch Exception" clauses in UFM's MQ code to prevent unhandled exceptions
  - Updated Shell Scripts (32-bit & 64-bit) to correctly handle the MQ_JAVA_LIB_PATH environment variable

- Universal File Mover v1.2.0
  - The Remote FileName, Directory and JobName values will now be stored as MQ property which means UFM requires MQ v7 or higher. Added MQSend attribute 'usev7prop' for backwards compatibility.
  - Added code to make sure ${LOOP_COUNTER} token is processed before other tokens.
  - Fixed a bug with tokens DATE+## and DATE-##
  - Fixed a bug with 'tofile' attribute processing for Copy, Convert, Move & ReEncodeAction
  - Fixed a bug with 'append' attribute for WriteText and MQReceive
  - UFM now requires MQ v7.0 or higher

- Universal File Mover v1.1.1
  - Fixed a bug with tokens DATE+## and DATE-##

- Added code to make sure ${LOOP_COUNTER} token is processed before other tokens.
- Fixed a bug with 'tofile' attribute processing for Copy, Convert, Move & ReEncodeAction
- Fixed a bug with 'append' attribute for WriteText and MQReceive

- Universal File Mover v1.1.0
  - Added Actions element to the MQReceive action
  - Added ReplyToQName element to MQ element in the Global section
  - Added ReplyToQName element to MQ element for MQSend action
  - Fixed a bug for Delete and Touch actions when they fail and onErrorFail is N.
  - Fixed a bug for MQSend action where it was requiring the file to exist at the start of the UFM Workflow.
  - Fixed a bug for Status queue manager name not being tokenized.

- Universal File Mover v1.0.5
  - Fixed a null pointer exception when Status element is not specified within the Global element.
  - Added a wait in MQReceive Action for the child thread to close the queue and disconnect from the queue manager when run as a daemon or Windows Service.

- Universal File Mover v1.0.4
  - Added MQ reconnection logic to MQReceive Action.
  - Added MQ reconnection logic to the StatusQueue functionality.
  - Updated the onerrorfail attribute for each action in the UFM_Workflow.dtd file to allow the onerrorfail attribute from the Global element to be used.

- Universal File Mover v1.0.3
  - Fixed an issue with the Action's checkObject not setting the error text in the GlobalErrorText.

- Universal File Mover v1.0.2
  - Fixed an issue with the MQReceive action when the same file is received again but is smaller than the original.

- Universal File Mover v1.0.1
  - Added new ReEncode Action to change the character encoding of a file
  - Fixed an issue with the MQReceive action using run='S' and putting an empty message to the backout queue.

- Universal File Mover v1.0.0
  - Initial release.

18 Appendix E – License Agreement

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this

definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

(a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

(b) You must cause any modified files to carry prominent notices stating that You changed the files; and

(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

(d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for

informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

Copyright 2012 Capitalware Inc.

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

19 Appendix F – Notices

Trademarks:

AIX, IBM, MQSeries, OS/2 Warp, OS/400, iSeries, MVS, OS/390, WebSphere, IBM MQ and z/OS are trademarks of International Business Machines Corporation.

HP-UX is a trademark of Hewlett-Packard Company.

Intel is a registered trademark of Intel Corporation.

Java, J2SE, J2EE, Sun and Solaris are trademarks of Sun Microsystems Inc.

Linux is a trademark of Linus Torvalds.

Mac OS X is a trademark of Apple Computer Inc.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation.

UNIX is a registered trademark of the Open Group.

WebLogic is a trademark of BEA Systems Inc.