



**Centers for Medicare & Medicaid Services  
CMS eXpedited Life Cycle (XLC)**

**Electronic Submission of Medical Documentation  
(esMD)**

**RC Client (Java) Implementation Guide**

---

**Version 1.5**

**09/30/2014**

**Document Number:** R\_3\_1\_RC\_Client\_Java\_Imp\_Guide

**Contract Number:** HHSM-500-2007-00024I

# Table of Contents

<b>1. Introduction</b>	<b>1</b>
<b>2. Overview</b>	<b>2</b>
<b>3. System Requirements</b>	<b>3</b>
3.1 Processor	3
3.2 Disk Space	3
3.3 Memory	3
3.4 Permissions	3
3.5 Network	3
3.6 Java	3
<b>4. TIBCO MFT File Transfers</b>	<b>4</b>
<b>5. Installation</b>	<b>5</b>
5.1 Out-of-the-box	5
5.1.1 KeyStore	5
5.1.2 Java Cryptography Extension (JCE) Policy Update	6
5.1.3 Configuring the RC Client	8
5.1.4 Running the RC Client	10
5.2 Custom RC Client	11
<b>6. Operation</b>	<b>12</b>
<b>7. XML Messages</b>	<b>13</b>
7.1 Inbound	13
7.1.1 Payload Files	13
7.1.2 Metadata File	13
7.1.3 Pickup HIH Status Response	14
7.1.4 PMD PA Review Result HIH Status Response	15
7.1.5 PMD PA Review Result Validation Error Response	15
7.1.6 Pickup Virus Scan Error Response	16
7.1.7 PMD PA Review Result Virus Scan Error Response	16
7.2 Outbound	17
7.2.1 Pickup Notification	17
7.2.2 Error Pickup Notification	18
7.2.3 PMD PA Review Result	18
<b>8. RC Client Components</b>	<b>20</b>
8.1 SFTP Client	20
8.2 Compression Utility	20
8.3 Encryption Utility	21
8.4 XML Processor	21
8.5 Scheduler	21

8.6	Housekeeping Manager .....	21
<b>9.</b>	<b>RC Client Workflow.....</b>	<b>22</b>
9.1	Start RC Client .....	24
9.1.1	Login and Encrypt.....	24
9.2	Outbound Process .....	24
9.2.1	Outbound Start .....	24
9.2.2	Get Outbound Documents .....	24
9.2.3	Connect .....	24
9.2.4	Push .....	24
9.3	Inbound Processes .....	25
9.3.1	Inbound Start .....	25
9.3.2	Housekeeping.....	25
9.3.3	Extraction.....	25
9.3.4	Checksum Verification .....	25
9.4	Acknowledgements .....	25
9.4.1	Pickup Notification .....	25
9.4.2	Error Pickup Notification .....	25
9.5	Connect.....	26
9.6	Get Notifications.....	26
9.7	Process Document.....	26
9.8	Pull Document.....	26
<b>10.</b>	<b>Release 3.1 Changes in the API.....</b>	<b>27</b>
<b>11.</b>	<b>Java Client API.....</b>	<b>30</b>
11.1	Security .....	30
11.2	Java API Documentation.....	30
11.2.1	Login.....	30
11.2.2	Inbound.....	31
11.2.3	Outbound.....	34
11.2.4	Utilities – PMDPA Result .....	35
11.2.5	Utilities - Encryption .....	36
11.2.6	Utilities - Handshake.....	38
11.3	Logs .....	38
11.4	Utilities.....	39
<b>12.</b>	<b>Error Codes .....</b>	<b>40</b>
<b>13.</b>	<b>Contact Information.....</b>	<b>41</b>
<b>Appendix A:</b>	<b>New User Registration.....</b>	<b>42</b>
<b>Appendix B:</b>	<b>Registered User Login Instructions.....</b>	<b>48</b>
<b>Appendix C:</b>	<b>XML Message Details.....</b>	<b>51</b>
<b>Acronyms.....</b>		<b>52</b>

<b>Glossary</b> .....	<b>53</b>
<b>Record of Changes</b> .....	<b>55</b>
<b>Approvals</b> .....	<b>56</b>

## List of Figures

Figure 1: RC Client Inbound and Outbound Process .....	2
Figure 2: Keystore Password Encryption .....	10
Figure 3: Private Key Password Encryption .....	10
Figure 4: RC Client Components .....	20
Figure 5: RC Client Workflow .....	23
Figure 6: Encryption and Decryption Process .....	30
Figure 7: New User Registration .....	42
Figure 8: New User Registration Menu .....	43
Figure 9: Terms and Conditions .....	43
Figure 10: New Registration Form .....	44
Figure 11: Email Verification .....	45
Figure 12: New Registration - Contact Information .....	45
Figure 13: Authentication Questions .....	46
Figure 14: Review Registration Details .....	46
Figure 15: Registration Acknowledgement.....	47
Figure 16: Registered User Login Window.....	48
Figure 17: Terms and Conditions .....	48
Figure 18: My Profile .....	49
Figure 19: Modify Account Profile .....	50

## List of Tables

Table 1: Inbound Files.....	4
Table 2: Outbound Files.....	4
Table 3: Keystore Creation Parameters.....	5
Table 4: JCE Policy files.....	7
Table 5: Java Development Kit.....	7
Table 6: Java Runtime Environment.....	7
Table 7: Security Directory.....	8
Table 8: Sample RC Client Configuration File.....	8
Table 9: E_123456-metadata.xml.....	13
Table 10: N_123456_Pickup_HIH_Status_Response.xml.....	14
Table 11: N_123456_PMDPA_Review_Result_HIH_Status_Response.xml.....	15
Table 12: R_123456_PMDPA_Review_Result_Validation_Error_Response.xml.....	15
Table 13: R_123456_Pickup_Virus_Scan_Error_Response.xml.....	16
Table 14: R_123456_PMDPA_Review_Result_Virus_Scan_Error_Response.xml.....	17
Table 15: P_186303_Pickup_Request.xml.....	17
Table 16: R_186303_Pickup_Error_Request.xml.....	18
Table 17: E_186303_PMDPA_Review_Result_Request.xml.....	18
Table 18: Client Method Comparison (Inbound).....	27
Table 19: Client Method Comparison (Outbound).....	28
Table 20: Login Details.....	31
Table 21: Inbound Method Details.....	32
Table 22: Retrieval of Outbound Documents Details.....	34
Table 23: esMD Manual Submit PMD PA Result.....	35
Table 24: Encryption.....	36
Table 25: Remote Troubleshooting.....	38
Table 26: RC Client Logs.....	39

Table 27: RC Client Utilities .....	39
Table 28: Error Codes .....	40
Table 29: Support Points of Contact.....	41
Table 30: XML Message Details .....	51
Table 31: Acronyms .....	52
Table 32: Glossary .....	53
Table 33: Record of Changes .....	55

# 1. Introduction

---

The Electronic Submission of Medical Documentation (esMD) system provides a mechanism for exchanging medical documentation (and responses) between the Medicare Provider community and the Medicare Review Contractor (RC) community. The purpose is to enable the electronic transmission of information between Health Information Handlers (HIHs) who represent Providers and the Medicare RCs, replacing paper documents where possible.

The RC Client is a utility that enables RCs to communicate with esMD by exchanging files via TIBCO® Managed File Transfer (MFT) Server. An example of a file is the Power Mobility Device (PMD) Prior Authorization (PA) review results response.

**Note:** esMD identifies submissions and requests sent from HIHs to RCs as inbound files, and identifies transactions and responses sent from RCs to HIHs as outbound files.

The RC Client provides the following functionality:

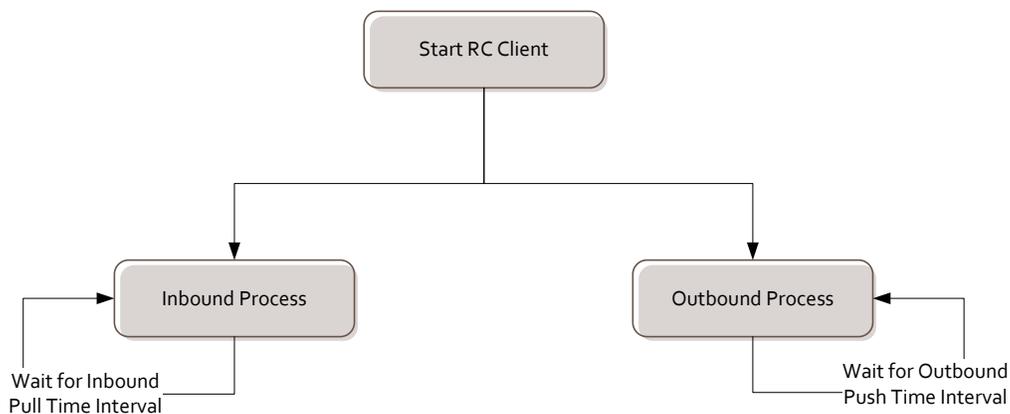
- Pull:
  - Inbound documents (submitted by HIHs) from the TIBCO MFT server;
  - HIH acknowledgements that they received PMD PA review results responses; and
  - Data Element Validations for the outbound process.
- Push:
  - Outbound PMD PA review results responses to esMD;
  - Error messages generated due to file decompression and checksum verification; and
  - Acknowledgement messages for receipt of documents and Authorization requests.
- Site-Specific Configuration Settings:
  - Push frequency/Pull frequency; and
  - Folder locations for both Inbound and Outbound files.

## 2. Overview

The esMD RC Client is a Java program that runs outside the Centers for Medicare & Medicaid Services (CMS) network on an RC's machine or server. The purpose of the RC Client is to connect to the CMS Baltimore Data Center (BDC) TIBCO MFT server and push and pull files. The RC Client uses the Individuals Authorized Access to the CMS Computer Services (IACS) login to authenticate the user's credentials through the TIBCO MFT server. The RC Client users (at the RC site) provide their login credentials when they start the RC Client on their machines. (See Appendix A: New User Registration for details on how to request an IACS Identification (ID).)

Users enter their login credentials only once at the program startup. When the RC Client starts, it initiates and then continuously runs two parallel threads as shown in Figure 1: RC Client Inbound and Outbound Process. When a user starts the RC Client, it will run continuously; it pulls and pushes files automatically without continual user intervention, based on the frequencies set by the RC.

Figure 1: RC Client Inbound and Outbound Process



In the inbound process when the RC Client connects to the BDC TIBCO MFT server, the RC Client immediately executes a pull cycle. The documents are pulled into the RC's inbound user directory for the authenticated user, and then the RC Client disconnects and waits for the next cycle, as determined by the Inbound Pull Time Interval setting.

In the outbound process when the RC Client connects to the BDC TIBCO MFT server, the RC Client executes a push cycle, pushes documents from the RC's outbound user directory to the TIBCO MFT server, and then disconnects and waits for the next cycle as determined by the Outbound Push Time Interval setting.

The inbound pull frequency is independent of the outbound push frequency. After each successful push or pull process, the RC Client thread disconnects from the TIBCO MFT server. To ensure continuous operation of the RC Client, it must preserve each user's IACS login credentials during the program execution.

## 3. System Requirements

---

### 3.1 Processor

The RC Client requires a Pentium 2 266-MHz processor or greater.

### 3.2 Disk Space

The disk requirement for the RC Java Client is 10 Megabytes (MB). The documents the RC Client pulls from the TIBCO MFT server may require additional disk space, as needed.

### 3.3 Memory

The RC Client requires a minimum of 50 MB of free memory.

### 3.4 Permissions

The RC Client must have read, write, and execute permissions on all the directories under the installation home.

### 3.5 Network

The RC Client requires internet connectivity that supports more than 32-Kilobits Per Second (Kbps) transfer speeds.

### 3.6 Java

The RC Client requires Java Runtime Environment (JRE) 1.6 or greater to run properly.

## 4. TIBCO MFT File Transfers

The RC Client uses a TIBCO MFT Server to interact with esMD. It uses the Secure Shell (SSH) File Transfer Protocol (SFTP) to connect to the TIBCO MFT Server and uses the Is/Get/Put commands to interact with the files. There are four (4) types of inbound files that RC Client pulls from the TIBCO MFT server, described in Table 1: Inbound Files.

**Note** “ES0001” is a sample mail box number that the TIBCO MFT Server uses to identify the RC and “0977890” is a sample transaction ID (also shown in Table 2: Outbound Files). The final two qualifiers in the file name that are prefixed with D and T are the Date and Timestamp, respectively. The VAL files will have ‘T’ prefix and the Production files will have ‘P’ prefix.

Only 1,022 files will be visible in the TIBCO MFT Server at one time by the MFT Mailbox Routing number. As each file is pulled, the TIBCO MFT Server will bring new files from the mainframe and place them at the bottom of the queue.

Table 1: Inbound Files

Type	Example File Name	Delivery Type Description
<b>Inbound</b>	T.ES0001.E0977890.D140116.T1033445	The E in prefix to the 0977890 transaction ID indicates an esMD payload
<b>Inbound</b>	T.ES0001.R0977890.D140116.T1033445	R indicates an esMD error
<b>Inbound</b>	T.ES0001.N0977890.D140116.T1033445	N indicates a notification file

Table 2: Outbound Files

Type	Example File Name	Delivery Type Description
<b>Outbound</b>	T#EFT.ON.ESMD.E0977890.D140116.T1033445	E indicates an esMD payload
<b>Outbound</b>	T#EFT.ON.ESMD.R0977890.D140116.T1033445	R indicates an esMD error
<b>Outbound</b>	T#EFT.ON.ESMD.P0977890.D140116.T1033445	P indicates an esMD pickup notification

## 5. Installation

You can install the RC Client in two ways:

- Out of the box; or
- Custom RC Client (Java).

### 5.1 Out-of-the-box

The RC Client Application Programming Interface (API) comes packaged with a sample client. To run the sample client out-of-the-box, the RC needs to follow the procedures in the following sections.

#### 5.1.1 KeyStore

##### 5.1.1.1 Creation

**Important: The RC Client uses asymmetric encryption to store the IACS user credentials securely. For this encryption to work, you need a secure Java KeyStore (JKS) with Public and Private keys of 2048 length. If you already have a JKS, you only need to update the configuration file with this information. Otherwise, follow the next procedures.**

1. If you do not have a JKS create one for the RC Client to use. (required).
2. Type the following command to create a new keystore for the RC Client to use.

```
keytool -genkey -keyalg RSA -keystore <keystore> -alias <alias> -storepass
<storepassword> -keypass <keypassword> -dname "CN=<commonName>,
OU=<organizationalUnit>, O=<organizationName>, L=<localityName>, S=<stateName>,
C=<country>" -keysize 2048 -validity 360
```

3. Replace <parameter> with the value of the parameter from the list in Table 3: Keystore Creation Parameters.

This command creates both the Public and Private keys using the Rivest, Shamir & Adleman (RSA) Algorithm with a key size of 2048 and validity of one year.

**Important: You must re-create these keys after they expire to continue to use the RC Client.**

Table 3: Keystore Creation Parameters

Where	Means
<keystore>	The keystore is the home location. If you do not specify this <code>keystore</code> option, the default keystore file named <code>.keystore</code> in the user's home directory will be created if it does not already exist. For example, <code>config/keystore.jks</code> .

Where	Means
<b>&lt;alias&gt;</b>	The certificate chain and the private key are stored in a new keystore entry identified by <i>alias</i> .
<b>&lt;storepassword&gt;</b>	The store password is used to protect the integrity of the keystore. It must be at least six characters long.
<b>&lt;keypassword&gt;</b>	The key password is used to protect the private key of the generated key pair. If no password is provided, the user is prompted for it. If you press Enter at the prompt, the key password is set to the same password as that used for the keystore. <i>Keypass</i> must be at least six characters long.
<b>&lt;commonName&gt;</b>	The common name is for any entity such as a person (for example, Susan Jones) or your company name.
<b>&lt;organizationalUnit&gt;</b>	The organizational unit can be used for a small organization, department, or division of an organization (for example, Purchasing).
<b>&lt;organizationName&gt;</b>	The organization name is for a large organization or company (for example, ABC Systems, Inc.).
<b>&lt;localityName&gt;</b>	The locality name can be for a city (for example, Palo Alto).
<b>&lt;stateName&gt;</b>	The state name can be for a U.S. state or province of another country (for example, California or Ontario in Canada).
<b>&lt;country&gt;</b>	The country is a two-letter code (for example, U.S.).

### 5.1.1.2 Integrity Verification

The command below will print the public key from the keystore and verify the keystore integrity.

1. Type the following command:

```
keytool -list -v -keystore <keystore> -storepass <storepassword> -alias <alias>
```

2. Replace <parameter> with the value for the parameter listed in Table 3: Keystore Creation Parameters.

## 5.1.2 Java Cryptography Extension (JCE) Policy Update

In addition to creating/providing the keystore, you may need to override the JCE security policy files if these files were not already overridden.

### 5.1.2.1 Understanding the JCE Security Policy Files

Due to import control restrictions, the version of the JCE security policy files that are bundled in the Java Development Kit™ (JDK) environment allow "strong" but limited cryptography to be used. To run the RC Client, this security policy must be overridden with the "unlimited strength" policy files that contain no restrictions on cryptographic strengths. If the RC Client is run with the default JCE security policy files, it will cause an error similar to the following:

```
java.security.InvalidKeyException: Illegal key size at javax.crypto.Cipher.a(DashoA13*...)
```

New JCE security policy files are packaged along with the RC Client and are in the "setup" subdirectory of the installation directory.

**Note:** These files do not contain additional encryption functionality because such functionality is supported in Sun's JDK.

### 5.1.2.2 Understanding the Export/Import Issues

JCE for JDK has been through the U.S. export review process. The JCE framework, along with the Sun JCE provider that comes standard with it, is exportable. The JCE architecture allows flexible cryptographic strength to be configured via jurisdiction policy files. Due to the import restrictions of some countries, the jurisdiction policy files distributed with the JDK software have built-in restrictions on available cryptographic strength.

### 5.1.2.3 JCE Policy Files

The setup directory in the RC Client installation contains the policy files listed in Table 4: JCE Policy files.

Table 4: JCE Policy files

Policy File	Description
<b>local_policy.jar</b>	Unlimited strength local policy file
<b>US_export_policy.jar</b>	Unlimited strength US export policy file

### 5.1.2.4 Installation Locations for Windows and Unix

<java-home> refers to the directory where the JRE was installed. It is determined based on whether you are running JCE on a JRE with the JDK installed. The JDK contains the JRE, but at a different level in the file hierarchy. Table 5: Java Development Kit and Table 6: Java Runtime Environment show examples of the installation for Java version 1.6, but this will work for Java version 7 as well.

Table 5: Java Development Kit

Environment	Example JDK Installation Directory	JAVA_HOME
<b>Windows</b>	C:\jdk1.6.0	C:\jdk1.6.0\jre
<b>Unix</b>	/home/user1/jdk1.6.0	/home/user1/jdk1.6.0/jre

Table 6: Java Runtime Environment

Environment	Example JRE Installation Directory	JAVA_HOME
<b>Windows</b>	C:\jre1.6.0	C:\jre1.6.0
<b>Unix</b>	/home/user1/jre1.6.0	/home/user1/jre1.6.0

#### Notes:

1. UNIX (Solaris/Linux) and Windows use different pathname separators; use the appropriate one ("\\" or "/") for your environment; and
2. On Windows, for each JDK installation, there may be an additional JRE installed under the "Program Files" directory. Ensure you install the unlimited strength policy JAR files for all JREs that you plan to use.

### 5.1.2.5 Setting Up Encryption/Decryption without Limitation

To use the encryption/decryption functionalities of the JCE framework without any limitation:

1. Make a copy of the original JCE policy files (US\_export\_policy.jar and local\_policy.jar in the standard place for JCE jurisdiction policy JAR files) in case you later decide to revert to these "strong" versions; and
2. Copy the policy files with the unlimited strength versions from the "setup" directory per the version of Java to be used (java6 or java7) under the installation directory to the security directory shown in Table 7: Security Directory.

Table 7: Security Directory

Environment	Installation Directory
Windows	<java-home>/lib/security
Unix	<java-home>\lib\security

### 5.1.3 Configuring the RC Client

Once the keystore is created and the policy files are installed, the RC Client is ready to be configured to use the keystore.

1. Update the keystore information in the configuration file. (Required); and  
**Important: The Extensible Markup Language (XML) configuration file (i.e., config/esmd-rc-client-config.xml) is used by the RC Client to retrieve important configuration parameters necessary for its operation.**
2. Use the comments for each configuration parameter shown in Table 8: Sample RC Client Configuration File as a guide in entering your data.

Table 8: Sample RC Client Configuration File

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:ESMDCConfig xmlns:ns2="http://esmd.ois.cms.hhs.gov/v1/rc/config"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance
xsi:schemaLocation="http://esmd.ois.cms.hhs.gov/v1/rc/config esmd-config.xsd
">

  <!--The TIBCO MFT Server Configuration-->
  <ESMDSFTPServer>
    <!--TIBCO MFT Sever host name or IP -->
    <host>eftp2.cms.hhs.gov</host>
    <!--The TIBCO MFT SFTP PORT-->
    <port>11022</port>
    <!-- Update: Use T for VAL, P for PROD-->
    <environmentId>T</environmentId>
    <!--The EFT File Name Prefix-->
    <eftFilePrefix>#EFT</eftFilePrefix>
  </ESMDSFTPServer>
  <!--The Keystore information for Encryption and Security-->
  <KeyStoreInfo>
    <!-- Update: The JKS Keystore Path-->
    <keyStoreLocation>/RCClient/config/keystore.jks</keyStoreLocation>
    <!-- Update: The Encrypted Keystore Password-->
    <encKeyInfo>ItwdafsdviaZNpvV54aRM9ZzQiw==</encKeyInfo>
```

```

    <!-- Update: The Encrypted Private Key Password-->
    <encKeyInfoExt>srs8adsfasRtLEB2I=</encKeyInfoExt>
    <!-- Update: The Certificate Alias-->
    <certAlias>selfsigned</certAlias>
  </KeyStoreInfo>
  <!--The Inbound Process Configuration-->
  <InboundConfig>
    <!-- Update: Enable the Inbound Process? true/false-->
    <enabled>true</enabled>
    <!--The Pull Frequency for the Inbound Process in minutes; the
default is 240 minutes i.e., 4 hours-->
    <checkFrequency>240</checkFrequency>
    <!-- Update: The RC Client installation/home directory-->
    <rcHomeDirectory>/RCClient</rcHomeDirectory>
    <!-- Update: The target directory to extract the downloaded inbound
files before routing-->
    <targetDirectory>/RCClient/data/download</targetDirectory>
    <!-- Update: The input directory where the inbound payloads and the
metadata will be routed after the extraction-->
    <inputDirectory>/RCClient/data/input</inputDirectory>
    <!-- Update: The temp directory where the files are pulled from TIBCO
MFT-->
    <tempDirectory>/RCClient/data/temp</tempDirectory>
    <!-- Update: The Error directory for routing the inbound error
notifications from esMD/HIH-->
    <errorDirectory>/RCClient/data/error</errorDirectory>
    <!-- Update: The configuration directory for RC Client-->
    <configDirectory>/RCClient/data/conf</configDirectory>
    <!-- Update: The notifications directory for routing the inbound
notifications from esMD/HIH-->
    <notificationsDirectory>/RCClient/data/notification</notificationsDirectory>
    <!-- Update: The Remote Inbound Directory path on the TIBCO MFT
Server-->
    <remoteInboundDir>/ES####</remoteInboundDir>
    <!--Update: The mail box number for the inbound files used to pick
the inbound files to pull-->
    <inboundRoutingId>ES####</inboundRoutingId>
  </InboundConfig>
  <!--The Outbound Process Configuration-->
  <OutboundConfig>
    <!-- Update: Enable the Outbound Process? true/false-->
    <enabled>true</enabled>
    <!--The push frequency for the Outbound process in minutes default is
15 minutes-->
    <pushFrequency>15</pushFrequency>
    <!-- Update: The temp directory to use for the outbound process for
creating the PMPDA/Notification files-->
    <tempDirectory>/RCClient/data/temp</tempDirectory>
    <!-- Update: The local outbound directory to push the outbound files
from-->
    <outputDirectory>/RCClient/data/output</outputDirectory>
    <!-- Update: The Remote Outbound directory to push files-->
    <remoteOutboundDir>/ES####_UPLOAD</remoteOutboundDir>
    <!--The Remote Outbound mail box number to push files onto esMD
servers via TIBCO MFT-->
    <outboundRoutingId>ESMD</outboundRoutingId>

```

```

    <!--The Outbound File name prefix-->
    <outboundFilePrefix>ON</outboundFilePrefix>
  </OutboundConfig>
  <!--The PMD PA Files Configuration-->
  <PMDPAConfig>
    <!--The Content Type Code for the PMDPA Files i.e., 8-->
    <contentTypeCode>8</contentTypeCode>
    <!--The Delivery Type for the PMDPA Files i.e.,E-->
    <deliveryType>E</deliveryType>
  </PMDPAConfig>
</ns2:ESMDCConfig>

```

### 5.1.3.1 Configuring Your Password Encryption

1. Run the encryptConfig.bat script to update the KeystoreInfo section with the encrypted keystore and private key password;
2. When the script prompts, enter your keystore and private key passwords, shown in Figure 2: Keystore Password Encryption and Figure 3: Private Key Password Encryption, and click **OK** in each Input window; and

Figure 2: Keystore Password Encryption



Figure 3: Private Key Password Encryption



3. Update the XML configuration file parameter "`certAlias`" with the alias of the certificate you created in Section 5.1.1.1 Creation.

The KeystoreInfo section of the XML Configuration file is now updated with the encrypted passwords and the certificate information required for the RC Client operation.

### 5.1.4 Running the RC Client

Before you, as the RC, run the sample RC Client, you must double-check all the configuration parameters in the XML configuration file, especially the ones with the "Update" prefix in the comments of the sample XML configuration file in Table 8: Sample RC Client Configuration File.

1. To run the sample RC Client, run the "rcclient.bat" utility provided in the distribution package; and
2. To stop the RC Client, stop the terminal window with a "Ctrl+C".

## 5.2 Custom RC Client

The RC Client provides an API so the RC can extend the RC Client to fit your environmental needs. The API enables you to perform the following functions:

- Log in to the TIBCO MFT server;
- Receive Notifications from the TIBCO MFT server using the SFTP protocol. (Refer to Section 11.2.2 below.);
- Decrypt/encrypt and store the login credentials using a secure RSA algorithm;
- Pull medical documentation from the TIBCO MFT server. (Refer to Section 11.2.2 below.);
- Extract the downloaded packages. (Refer to Section 11.2.2 below.);
- Check the payloads using checksums in the metadata. (Refer to Section 11.2.2 below.);
- Create custom files (for example, custom PMD PA files). (Refer to Section 11.2.3 below.); and
- Push the outbound files from the “output” directory. (Refer to Section 11.2.3 below.)

**Note:** The procedures for customizing the RC Client API are beyond the scope of this document; they are in the Java Documentation (Javadoc) packaged in the RC Client.

## 6. Operation

---

The RC Client runs in a cyclical manner, sleeping for a specified time interval between the operating cycles. The sleep intervals are configured in the “checkFrequency” parameter for the Inbound Process and the “pushFrequency” parameter for the Outbound process. The RC is advised to use the default of 240 minutes (4 hours) for the Inbound process and 15 minutes for the Outbound process.

The RC Client operation is interrupted in two scenarios:

1. The IACS Passwords expire (they expire every 60 days; contact the CMS Help Desk to reset); and
2. Virus Scan error notification is received from esMD.

In the first scenario, when the IACS Password expires; the RC Client suspends its operation and is terminated. The RC must restart the RC Client and the user must provide the right credentials to login to the TIBCO MFT Server. The IACS notifies the user 15 days prior to the password expiring.

In the second scenario, when a Virus Scan error notification has been received from esMD, all the processes of the RC Client are suspended and the RC Client is terminated. In addition, the RC Client is locked and cannot pull/push files even if the RC Client is restarted. The RC is advised to contact the esMD team (refer to Section 13 Contact Information for more details) to unlock the RC Client.

## 7. XML Messages

This section describes the various XML messages transferred during the inbound and outbound processes.

**Note:** Refer to Appendix C: XML Message Details for details on the name of the XML Message Files transferred between the RC Client and esMD.

### 7.1 Inbound

The RC Client transfers the following files during the inbound process:

- Payload Files;
- Metadata File;
- Pickup HIH Status Response;
- PMD PA Review Result HIH Status Response;
- PMD PA Review Result Validation Error Response;
- Pickup Virus Scan Error Response; and
- PMD PA Review Result Virus Scan Error Response.

#### 7.1.1 Payload Files

The RC Client will receive Portable Document Format (PDF) files as payloads in the inbound documents with delivery type 'E'. An example payload file name is E\_185457-esmdQSSIVG0407141396893280928-0.pdf.

#### 7.1.2 Metadata File

The metadata file accompanies the payload files in the inbound documents with delivery type 'E'. The metadata file contains information about the payloads like the Object Identifier (OID), Transaction ID, Submission metadata, and optional metadata. The Content Type Code will change for each line of business. See Table 9: E\_123456-metadata.xml.

**Note:** The metadata file will remain the same for all lines of business including Additional Documentation Requests (ADRs), PMD PA Requests, Non-Emergent Ambulance Transport and Hyperbaric Oxygen (HBO) Prior Authorization requests Appeals, and Advance Determination of Medicare Coverage (ADMCs).

**Note:** As of Release 3.1, the Claim ID is optional for Appeals.

Table 9: E\_123456-metadata.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:RetrieveMedicalDocumentationResponse returnCode="0"
serviceSuccessful="true" xmlns:ns2="http://esmd.ois.cms.hhs.gov/v1/rc">
  <statusDescription>The RetrieveMedicalDocumentationRequest processed
successfully.</statusDescription>
  <NumberOfDocuments>1</NumberOfDocuments>
  <ESMDPackage>
    <ESMDTransaction TransactionId="123456" DeliveryType="E"/>
  </ESMDPackage>
</ns2:RetrieveMedicalDocumentationResponse>
```

```

    <SendingOID>urn:oid:123.456.657.162</SendingOID>
    <TargetOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.1</TargetOID>
    <CompleteSubmission>true</CompleteSubmission>
    <SubmissionMetadata xsi:type="ns4:ADR"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns4="http://esmd.ois.cms.hhs.gov/v1/rc/cmsbt">
        <CreationTime>2014-04-25T12:21:32.608-04:00</CreationTime>
        <SubmissionTime>2014-04-25T12:21:32.608-04:00</SubmissionTime>
        <EFTSubmissionTime>2014-04-25T12:21:32.609-
04:00</EFTSubmissionTime>
        <ContentTypeCode>1</ContentTypeCode>
        <NPI>1234567890</NPI>
        <ClaimId>8071302</ClaimId>
        <CaseId>123</CaseId>
    </SubmissionMetadata>
    <Documentation DocumentUniqueIdentifier="E_537143-
esmdQSSISRADRVALID1398442888298-0" MimeType="application/pdf">
        <OptionalMetadata>
            <FieldName>esMDDocumentCreationTime</FieldName>
            <FieldValue>1398442892609</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>Description</FieldName>
            <FieldValue>From esMD</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>CheckSum</FieldName>
            <FieldValue>73d1ba48402985bac6ddab12f47c179dddbbe4c6</FieldValue>
        </OptionalMetadata>
    </Documentation>
</ESMDPackage>
</ns2:RetrieveMedicalDocumentationResponse>

```

### 7.1.3 Pickup HIH Status Response

When the RC Client sends a pickup notification to esMD, the esMD application processes the notification and sends the response to the HIH. The esMD application then generates the Pickup Status Response and sends it to the RC, indicating the response was sent to the HIH, as detailed in the code in Table 29: XML Message Details in Appendix C:

N\_TID\_Pickup\_HIH\_Status\_Response.xml.

**Note:** The HIH Pickup Status Response will remain the same for all lines of business including ADRs, Appeals, PMD PA Responses, Recovery Audit Contractor (RAC) Discussion Requests, and ADMC Requests.

Table 10: N\_123456\_Pickup\_HIH\_Status\_Response.xml

```

<?xml version="1.0" encoding="UTF-8" standalone="true"?>
<ns2:RCPickupNotificationResponse
xmlns:ns2="http://esmd.ois.cms.hhs.gov/v1/rc/config">
    <ESMDTransactionId>123456</ESMDTransactionId>
    <ErrorInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:nil="true"/>

```

```

<status>Success</status>
<statusDesc>SENT PICKUP STATUS TO HIH</statusDesc>
</ns2:RCPickupNotificationResponse>

```

### 7.1.4 PMD PA Review Result HIH Status Response

When the RC Client sends a PMD PA Review Result to esMD, the esMD application processes the file and sends the PMD PA Review Result to the HIH. The esMD application submits the PMD PA Review Result HIH Status Response, detailed in Table 11:

N\_123456\_PMDPA\_Review\_Result\_HIH\_Status\_Response.xml, and sends it to the RC, indicating the result was sent to the HIH. Please refer to the code located in Appendix C, Table 29 File N\_TID\_PMDPA\_Review\_Result\_HIH\_Status\_Response.xml.

Table 11: N\_123456\_PMDPA\_Review\_Result\_HIH\_Status\_Response.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<esmd:SubmitPMDPADeterminationResponse
xsi:schemaLocation="http://esmd.ois.cms.hhs.gov/v1/rc
../../../../config/esmd-rc.xsd "
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:esmd1="http://esmd.ois.cms.hhs.gov/v1/rc/transaction"
xmlns:esmd="http://esmd.ois.cms.hhs.gov/v1/rc"
xmlns:cmsbt="http://esmd.ois.cms.hhs.gov/v1/rc/cmsbt">
  <statusDescription>PMD PA Review results - Successfully delivered
to HIH</statusDescription>
  <ESMDTransaction DeliveryType="N" TransactionId="123456"/>
</esmd:SubmitPMDPADeterminationResponse>

```

### 7.1.5 PMD PA Review Result Validation Error Response

When the RC Client sends a PMD PA Review Result to esMD, the esMD application processes and sends the PMD PA Review Result to the HIH. If there is an error in processing the PMD PA Review Result submitted by the RC, the esMD application generates the PMD PA Results Response Error, detailed in Table 12:

R\_123456\_PMDPA\_Review\_Result\_Validation\_Error\_Response.xml, and sends it to the RC. The RC will then resubmit the PMD PA Results Result. Please refer to the code located in Appendix C, Table 29: XML Message Details file, R\_TID\_PMDPA\_Review\_Result\_Validation\_Error\_Response.xml.

Table 12: R\_123456\_PMDPA\_Review\_Result\_Validation\_Error\_Response.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<esmd:SubmitPMDPADeterminationResponse
xsi:schemaLocation="http://esmd.ois.cms.hhs.gov/v1/rc/config/esmd-
rc.xsd "
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:esmd1="http://esmd.ois.cms.hhs.gov/v1/rc/transaction"
xmlns:esmd="http://esmd.ois.cms.hhs.gov/v1/rc"
xmlns:cmsbt="http://esmd.ois.cms.hhs.gov/v1/rc/cmsbt">
  <statusDescription>statusDescription</statusDescription>

```

```

<ESMDTransaction DeliveryType="R" TransactionId="123456"/>
<ValidationFailure>
  <FailureCode>541</FailureCode>
  <FailureReason>ESMD validation error: Transaction ID is
invalid</FailureReason>
</ValidationFailure>
<ValidationFailure>
  <FailureCode>556</FailureCode>
  <FailureReason>ESMD validation error: Decision Indicator must
be A, N, or R</FailureReason>
</ValidationFailure>
</esmd:SubmitPMDPADeterminationResponse>

```

### 7.1.6 Pickup Virus Scan Error Response

When the RC Client sends a Pickup Notification to esMD, the esMD application sends it to the Virus Scan Gateway for virus scan. If there are any viruses detected in the pickup notification, the esMD application sends the message detailed in Table 13:

R\_123456\_Pickup\_Virus\_Scan\_Error\_Response.xml to the RC. The RC Client will then pull this Virus Scan Error, stop the inbound and outbound processes and lock down the RC Client to prevent RC Client from interacting with esMD. In this situation, the RC Client does not enable recovery, and the RC will contact esMD Help Desk. Refer to the code located in Table 29: XML Message Details in Appendix C: R\_TID\_Pickup\_Virus\_Scan\_Error\_Response.xml.

Table 13: R\_123456\_Pickup\_Virus\_Scan\_Error\_Response.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<tns:RCPickupNotificationResponse
xmlns:tns="http://esmd.ois.cms.hhs.gov/v1/rc/config"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://esmd.ois.cms.hhs.gov/v1/rc/config esmd-
config.xsd">
  <ESMDTransactionId>123456</ESMDTransactionId>
  <ErrorInfo>
    <ErrorCode>560</ErrorCode>
    <ErrorName>VirusFound</ErrorName>
    <ErrorDescription>ESMD validation error: Submission is infected with
virus</ErrorDescription>
  </ErrorInfo>
  <Status>FAILED</Status>
  <StatusDesc>Outbound Response File contains virus and so the response is
rejected.</StatusDesc>
</tns:RCPickupNotificationResponse>

```

### 7.1.7 PMD PA Review Result Virus Scan Error Response

When the RC Client sends a PMD PA Review Result to esMD, the esMD application sends it to the Virus Scan Gateway for virus scan. If there are any viruses detected in the PMD PA Review Result, the esMD application sends the message detailed in Table 14:

R\_123456\_PMDPA\_Review\_Result\_Virus\_Scan\_Error\_Response.xml to the RC. The RC Client will then pull this PMD PA Review Result Virus Scan Error, stop the inbound and outbound processes and lock down the RC Client to prevent RC Client from interacting with esMD. In this situation, the RC Client does not enable recovery, and the RC will contact esMD

Help Desk. Refer to the code located in Appendix C, Table 29: XML Message Details File R\_TID\_PMDPA\_Review\_Result\_Validation\_Error\_Response.xml.

Table 14: R\_123456\_PMDPA\_Review\_Result\_Virus\_Scan\_Error\_Response.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<esmd:SubmitADMCDeterminationResponse
xmlns:cmsbt="http://esmd.ois.cms.hhs.gov/v1/rc/cmsbt"
xmlns:esmd="http://esmd.ois.cms.hhs.gov/v1/rc"
xmlns:esmdl="http://esmd.ois.cms.hhs.gov/v1/rc/transaction"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://esmd.ois.cms.hhs.gov/v1/rc ../config/esmd-rc.xsd
">
  <statusDescription>Outbound Response File contains virus and so the
response is rejected.</statusDescription>
  <ESMDTransaction TransactionId="123456" DeliveryType="R"/>
  <ValidationFailure>
    <FailureCode>560</FailureCode>
    <FailureReason>ESMD validation error: Submission is infected with
virus</FailureReason>
  </ValidationFailure>
</esmd:SubmitADMCDeterminationResponse>
```

## 7.2 Outbound

The RC Client transfers the following messages during the outbound process:

- Pickup Notification;
- Error Pickup Notification; and
- PMD PA Review Result.

### 7.2.1 Pickup Notification

The RC Client generates pickup notifications for all inbound files with delivery type “E” pulled from TIBCO MFT Server and processed successfully, as detailed in Table 15: P\_186303\_Pickup\_Request.xml. Please refer to the code located in Appendix C, Table 29: XML Message Details, File P\_TID\_Pickup\_Request.xml.

Table 15: P\_186303\_Pickup\_Request.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<tns:RCPickupNotification
xmlns:tns="http://esmd.ois.cms.hhs.gov/v1/rc/config"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://esmd.ois.cms.hhs.gov/v1/rc/config/esmd-
config.xsd ">
  <ESMDTransactionId>186303</ESMDTransactionId>
  <PickupTime>2014-04-09T09:00:00</PickupTime>
  <SubmissionTime>2014-04-09T09:00:00</SubmissionTime>
  <ErrorInfo xsi:nil="true"/>
</tns:RCPickupNotification>
```

## 7.2.2 Error Pickup Notification

The RC Client generates pickup error notifications for all inbound files pulled from TIBCO MFT and processed unsuccessfully, as detailed in Table 16: R\_186303\_Pickup\_Error\_Request.xml. The processing errors are generated in two scenarios:

- Checksum verification failed (i.e., the payload file received by the RC client does not match the file sent by esMD); and
- Extraction was unsuccessful (i.e., the RC client could not successfully unzip the file received from the server).

Please refer to the code located in Appendix C, Table 29: XML Message Details, File R\_TID\_Pickup\_Error\_Request.xml.

Table 16: R\_186303\_Pickup\_Error\_Request.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<tns:RCPickupNotification
  xmlns:tns="http://esmd.ois.cms.hhs.gov/v1/rc/config"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://esmd.ois.cms.hhs.gov/v1/rc/config/esmd-
  config.xsd ">
  <ESMDTransactionId>186303</ESMDTransactionId>
  <PickupTime>2001-12-31T12:00:00</PickupTime>
  <SubmissionTime xsi:nil="true"/>
  <ErrorInfo>
    <ErrorCode>0</ErrorCode>
    <ErrorName>ErrorName</ErrorName>
    <ErrorDescription>ErrorDescription</ErrorDescription>
  </ErrorInfo>
</tns:RCPickupNotification>
```

## 7.2.3 PMD PA Review Result

The PMD PA Review Result is the XML message from the RC to the HIH, to inform the HIH of the decision as detailed in Table 17: E\_186303\_PMDPA\_Review\_Result\_Request.xml. Refer to the code located in Appendix C, Table 29: XML Message Details, File E\_TID\_PMDPA\_Review\_Result\_Request.xml.

**Note:** If the Denial Code description field is present in the Review Result Response to the HIH, the esMD system will remove the Denial Code description before sending the Review Result Response to the HIH.

Table 17: E\_186303\_PMDPA\_Review\_Result\_Request.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<esmd:SubmitPMDPADeterminationRequest
  xmlns:cmsbt="http://esmd.ois.cms.hhs.gov/v1/rc/cmsbt"
  xmlns:esmd="http://esmd.ois.cms.hhs.gov/v1/rc"
  xmlns:esmdl="http://esmd.ois.cms.hhs.gov/v1/rc/transaction"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

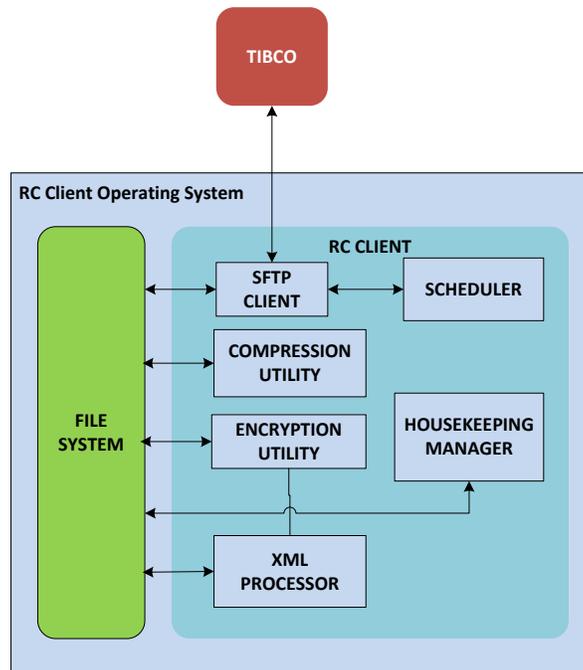
```
xsi:schemaLocation="http://esmd.ois.cms.hhs.gov/v1/rc/config/esmd-rc.xsd ">
  <ESMDTransaction TransactionId="186303" DeliveryType="E"/>
  <PMDPAResponse>
    <CreationTime>2014-04-09T10:00:00</CreationTime>
    <SubmissionTime>2014-04-09T10:00:00</SubmissionTime>
    <EFTSubmissionTime>2014-04-09T09:00:00</EFTSubmissionTime>
    <ContentTypeCode>8</ContentTypeCode>
    <NPI>1234567890</NPI>
    <DecisionIndicator>A</DecisionIndicator>

  <UniqueTrackingNumber>UniqueTrackingNumber</UniqueTrackingNumber>
    <ReasonCodeRecord>
      <ReasonCode>123</ReasonCode>
      <ReasonCodeDescription>test123</ReasonCodeDescription>
    </ReasonCodeRecord>
  </PMDPAResponse>
</esmd:SubmitPMDPADeterminationRequest>
```

## 8. RC Client Components

Figure 3: RC Client Components shows the internal components of RC Client application. The following sections describe each component in detail.

Figure 4: RC Client Components



### 8.1 SFTP Client

The SFTP Client is an internal component of the RC Client application. It provides the following functionality:

- Connect to the TIBCO MFT server using IACS ID;
- List the available documents on the TIBCO MFT server;
- Pull the documents down to the RC Client; and
- Push the outbound documents from RC Client to the TIBCO MFT server.

### 8.2 Compression Utility

The Compression utility allows the RC Client to extract the payload, metadata file, and messages from the compressed file downloaded from the TIBCO MFT server. The RC Client uses the zip file format.

The same utility is used to create compressed file logs for extraction.

### 8.3 Encryption Utility

The Encryption utility encrypts the login credentials that will be stored in memory for the duration of the RC Client program execution. The Encryption utility is described in detail in Section 11.1 Security.

### 8.4 XML Processor

The XML Processor supports creating XML messages to send to esMD, as well as loading the configuration files for the RC Client.

### 8.5 Scheduler

After the RC Client starts, the polling cycle begins. The poll is a redundant cycle; you can configure the interval (for example, 1 hour or 4 hours) through the RC Client property file. The Schedule component controls the RC Client threads and ensures the RC Client runs at regular intervals determined by the checkFrequency parameter in the XML Configuration File.

### 8.6 Housekeeping Manager

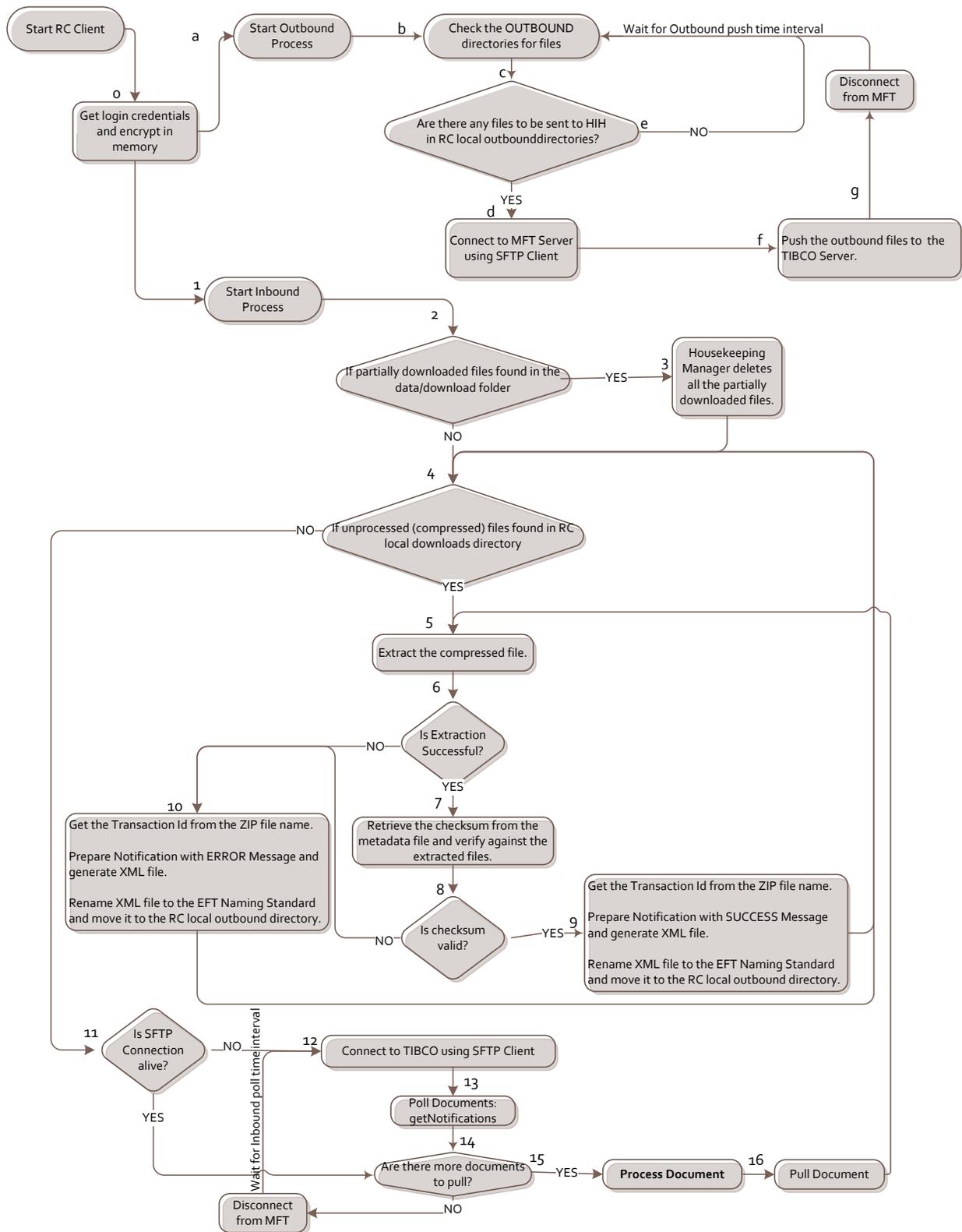
The Housekeeping Manager allows the RC Client to recover from any abnormal terminations, with the exception of a Virus lockdown. In this situation, the RC Client does not enable recovery, and the RC will contact esMD Help Desk.

## 9. RC Client Workflow

---

The workflow associated with Figure 4: RC Client Components is broken down in Figure 5: RC Client Workflow, followed by a detailed description of the workflow.

Figure 5: RC Client Workflow



## 9.1 Start RC Client

The RC Client starts on the RC machine or server. It loads the XML Configuration File.

### 9.1.1 Login and Encrypt

The RC Client prompts the user for the following details:

1. IACS User ID; and
2. IACS Password.

After successful login, TIBCO MFT Server login credentials are encrypted in memory and used when needed to log in to the TIBCO MFT server. The RC Client initiates two threads, one for the outbound process and one for the inbound process in sections 9.2 Outbound Process and 9.3 Inbound Processes, respectively.

## 9.2 Outbound Process

### 9.2.1 Outbound Start

The RC Client loads configuration parameters for the outbound process from the configuration file (XML). The configuration parameters are as follows:

- Directories used by the RC Client to create the outbound files (`outputDirectory`);
- The remote outbound directory to push the files to (`remoteOutboundDir`);
- Push frequency (`pushFrequency`);
- The outbound file name prefix for the TIBCO MFT server (`outboundFilePrefix`); and
- SFTP server details for the chosen environment (`ESMDSFTPServer`).

### 9.2.2 Get Outbound Documents

The RC Client checks the output directory for any files to be sent to the HIIH. If any such files exist, the process continues to Step d (Connect); otherwise, the outbound process thread sleeps for the time interval determined by the pushFrequency parameter in the XML Configuration File.

### 9.2.3 Connect

The RC Client connects to the TIBCO MFT server using IACS login credentials. The Encryption utility decrypts the credentials in memory and logs in to the TIBCO MFT server. If the user password is expired, the connection fails, prompting the user to provide the login information again.

### 9.2.4 Push

The RC Client pushes outbound files to the TIBCO MFT server. After that, the outbound process thread sleeps. The sleep time interval is determined by the outbound push frequency configuration parameter in the XML Configuration File.

## 9.3 Inbound Processes

### 9.3.1 Inbound Start

The RC Client loads Configuration parameters from the configuration file (XML). The Configuration parameters are for the following inbound processes:

- Directories used by the Inbound process;
- Pull frequency; and
- SFTP server details for the chosen environment.

### 9.3.2 Housekeeping

### 9.3.3 Extraction

The Housekeeping Manager extracts compressed files found in the local “temp” directory for the RC Client before it pulls any new documents from the TIBCO MFT server. It will extract the oldest files first. If the extraction is successful, the RC Client proceeds to “checksum verification”; otherwise, the RC Client creates an error pickup notification.

### 9.3.4 Checksum Verification

After the extraction is complete, the RC Client uses the XML Processor to parse the metadata file from the zip package. The metadata file contains the checksums for all payloads in the package. The RC Client verifies the checksum for each file in the package against the checksum in the metadata file. If the checksum is valid for all files, the RC Client will create a pickup notification; otherwise, the RC Client will create an error pickup notification.

## 9.4 Acknowledgements

### 9.4.1 Pickup Notification

If the RC Client successfully extracts and verifies compressed files, the RC Client sends a success notification through esMD to inform the HIH that the document has been received and successfully processed. To generate this success notification, the RC Client should:

- Obtain the transaction ID from the compressed file name;
- Prepare the notification with a SUCCESS message and generate an XML notification file; and
- Rename the XML notification file to the Enterprise File Transfer (EFT) system naming standard and move it to the outbound directory. Refer to Section 9.2 Outbound Process for more information.

### 9.4.2 Error Pickup Notification

If the RC Client encounters an error indicating failure while either extracting the compressed file or verifying the checksum for the contents of the package, the RC Client sends an error

notification to through esMD, asking the HIH to resubmit the package. In order to generate this error notification, the RC Client must:

- Obtain the transaction ID from the compressed file name;
- Prepare the notification with an ERROR message;
- Generate an XML notification file; and
- Rename the XML notification file to the EFT naming standard and move it to the outbound directory. This file will be handled by the outbound process.

## 9.5 Connect

After the Housekeeping Manager completes preprocessing, the RC Client checks for an active connection to the TIBCO MFT server. If a connection is active, the RC Client uses this connection. If the connection is inactive, the RC Client uses the Encryption utility to decrypt the login credentials from memory and connects to the TIBCO MFT server.

## 9.6 Get Notifications

The RC Client uses the SFTP Client to get a list of the available inbound documents for the RC on the TIBCO MFT server.

## 9.7 Process Document

If any documents are available for the RC Client to pull from the TIBCO MFT server, the RC Client will go through the list to pull each document.

## 9.8 Pull Document

The RC Client uses the SFTP Client to pull each inbound document from the TIBCO MFT server. The RC Client then extracts the contents of the zip file and continues processing.

## 10. Release 3.1 Changes in the API

Table 18: Client Method Comparison compares similar methods in the Enterprise Content Management (ECM) Client and the Release 3.1 RC Client.

Table 18: Client Method Comparison (Inbound)

ECM Client (Inbound)	
<p><b>ESMDDownloadDocumentation.getNotifications()</b></p> <ul style="list-style-type: none"> <li>Creates a "RetrieveNotificationsRequest" with the contentTypeCode and deliveryType.</li> <li>Retrieves the list of ESMDTransactions available.</li> </ul>	<p>InboundProcessImpl.getNotifications()</p> <ul style="list-style-type: none"> <li>Connects to TIBCO MFT server with IACS Login and Password.</li> <li>Retrieves the list of files available for download for that environment.</li> </ul>

ECM Client (Inbound)	The RC Client (Inbound)
<p><b>ESMDDownloadDocumentation.processMedicalDocumentation()</b></p> <ul style="list-style-type: none"> <li>• Creates a “RetrieveMedicalDocumentation Request” from the ESMDTransaction.</li> <li>• Retrieves the Medical Document for that transaction in “RetrieveMedicalDocumentation Response”.</li> <li>• Calls saveDocuments() method to save the metadata from the response and the attachments to the target directory from the XML configuration file.</li> </ul>	<p>InboundProcessImpl.processMedicalDocumentation()</p> <ul style="list-style-type: none"> <li>• Extracts the zip file into the “download” directory using the extractDocument() method.</li> <li>• If extraction fails, calls the acknowledge method with an error event and exits.</li> <li>• After successful extraction, verifies the extracted payloads against the checksum in the metadata file using the checkPayloads() method.</li> <li>• If checksum fails, calls the acknowledge method with an error event.</li> <li>• If checksum passes, calls the acknowledge method with a success event.</li> <li>• Routes the inbound files to the local directories. Refer to Appendix C:XML Message Details for more details on the routing rules.</li> </ul>
<p><b>ESMDDownloadDocumentation.acknowledge()</b></p> <ul style="list-style-type: none"> <li>• Creates and submits the “SubmitExternalEventRequest” for the ESMDTransaction passed with eventCode 100.</li> <li>• Retrieves the “SubmitExternalEventResponse ” and logs it.</li> </ul>	<p>InboundProcessImpl.acknowledge()</p> <ul style="list-style-type: none"> <li>• Creates the error/success message and puts it in the output directory.</li> <li>• Logs the event.</li> </ul>

Table 19: Client Method Comparison (Outbound)

ECM Client (Outbound)	RC Client (Outbound)
<p><b>ESMDManualSubmitADMCResult.promptForInfo()</b></p> <ul style="list-style-type: none"> <li>• Gathers the following information                             <ul style="list-style-type: none"> <li>○ Delivery Type</li> <li>○ Transaction ID</li> <li>○ Content Type Code</li> <li>○ NPI</li> <li>○ ADMC Decision Indicator</li> <li>○ Unique Tracking Number (enter for none)</li> <li>○ Add a reason code Y/N</li> <li>○ Reason Code</li> <li>○ Reason Code Description (enter for none)</li> </ul> </li> </ul>	<p><b>ESMDManualSubmitPMDPAResultImpl.promptForInfo()</b></p> <ul style="list-style-type: none"> <li>• Gathers the following information                             <ul style="list-style-type: none"> <li>○ Delivery Type</li> <li>○ Transaction ID</li> <li>○ Content Type Code</li> <li>○ NPI</li> <li>○ PMDPA Decision Indicator</li> <li>○ Unique Tracking Number (enter for none)</li> <li>○ Add a reason code Y/N</li> <li>○ Reason Code</li> <li>○ Reason Code Description (enter for none)</li> </ul> </li> </ul>

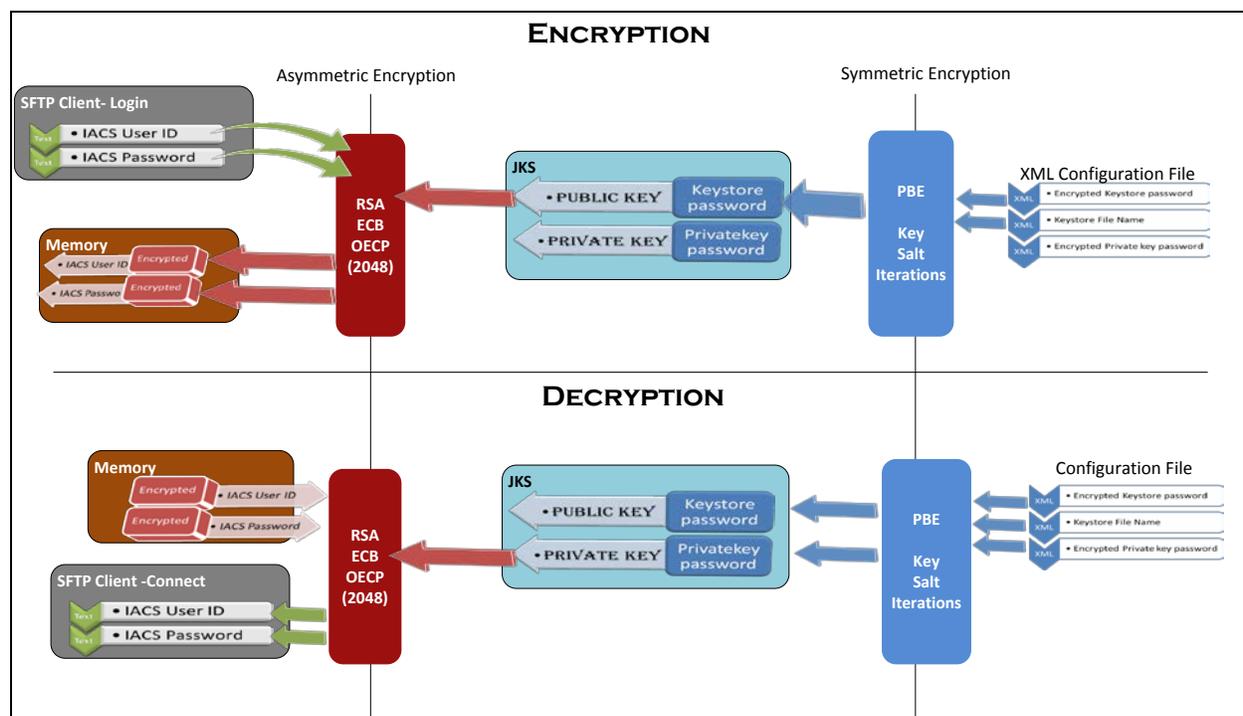
ECM Client (Outbound)	RC Client (Outbound)
<ul style="list-style-type: none"> <li>Builds the ADMCResult and the ESMDTransaction needed for submission.</li> </ul>	<ul style="list-style-type: none"> <li>Builds the PMDPAResult object.</li> </ul>
<p><b>ESMDManualSubmitADMCResult.submitADMCResult()</b></p> <ul style="list-style-type: none"> <li>Creates and submits the “SubmitADMCDeterminationRequest”.</li> <li>Retrieves the “SubmitADMCDeterminationResponse” and logs it.</li> </ul>	<p>ESMDManualSubmitPMDPAResultImpl.submitPMDPAResult()</p> <ul style="list-style-type: none"> <li>Takes the PMDPAResult object and creates the XML response file in the input directory.</li> <li>Compresses and pushes the XML file to TIBCO MFT server.</li> </ul>

# 11. Java Client API

## 11.1 Security

When the RC Client starts, the user credentials are provided because they are stored in encrypted form in memory. Figure 6: Encryption and Decryption Process shows the process of encryption and decryption to safeguard the IACS user credentials from exposure.

Figure 6: Encryption and Decryption Process



The RC Client uses a combination of Symmetric (PBEWithMD5AndTripleDES) and Asymmetric (RSA/ECB/OAEPWithSHA1AndMGF1Padding) Encryption Algorithms to secure the login credentials.

## 11.2 Java API Documentation

This section discusses API methods that can be called for a custom solution to interface with TIBCO MFT Server. If you, as the RC, choose to use the RC Client out-of-the-box, skip this section.

### 11.2.1 Login

Table 20: Login Details lists the methods and their descriptions used in the login process.

Table 20: Login Details

#	Method	Description
1.	public LoginDetails loginAndEncrypt(SftpDetails sftpDetails_, ESMDCConfig.KeyStoreInfo keyStoreInfo_) throws Exception;	Logs into the server and stores the encrypted login information.  Parameters: <ol style="list-style-type: none"> <li>sftpDetails_ – The SFTP server Details</li> <li>keyStoreInfo_ – The Keystore Details</li> </ol> Returns: The LoginDetails Object with the following properties populated: <ol style="list-style-type: none"> <li>encryptedUID – Encrypted User ID</li> <li>encryptedPWD – Encrypted Password</li> </ol>
2.	public LoginDetails decryptAndLogin(LoginDetails loginDetails_, SftpDetails sftpDetails_, ESMDCConfig.KeyStoreInfo keyStoreInfo_) throws Exception;	Decrypts the login credentials passed in the LoginDetails object and logs into the TIBCO MFT server.  Parameters: <ol style="list-style-type: none"> <li>loginDetails_ – the LoginDetails object with the following properties populated: <ul style="list-style-type: none"> <li>encryptedUID – Encrypted User ID</li> <li>encryptedPWD – Encrypted Password</li> </ul> </li> <li>sftpDetails_ – The SFTP server Details</li> <li>keyStoreInfo_ – The Keystore Details</li> </ol> Returns: The LoginDetails Object with the following properties populated: <ol style="list-style-type: none"> <li>encryptedUID – Encrypted User Id.</li> <li>encryptedPWD – Encrypted Password.</li> <li>channelSftp – SFTP Channel connected.</li> </ol>

## 11.2.2 Inbound

Table 21: Inbound Method Details lists the methods and their descriptions used in the inbound process.

Table 21: Inbound Method Details

#	Method	Description
1.	<pre>public List&lt;String&gt; getNotifications(LoginDetails loginDetails_,String remoteDownloadDirectoryPath_, String filePattern_) throws SftpException ;</pre>	<p>Uses the LoginDetails object to list the remote directory.</p> <p>Parameters:</p> <ol style="list-style-type: none"> <li>1. loginDetails_ – the LoginDetails object with the following properties populated: <ul style="list-style-type: none"> <li>• encryptedUID – Encrypted User Id</li> <li>• encryptedPWD – Encrypted Password.</li> <li>• channelSftp – SFTP Channel connected.</li> </ul> </li> <li>2. remoteDownloadDirectoryPath_ – The remote directory path to download from as a String.</li> <li>3. filePattern_ – The File Name Pattern to look for as a String.</li> </ol> <p>Returns: The List&lt;String&gt; with the filenames to pull.</p>
2.	<pre>public void pullDocument(String remoteDocumentName_, String localDocumentName_, LoginDetails loginDetails_) throws Exception;</pre>	<p>Pulls the document (namely, the zip file) from the TIBCO MFT server with the name remoteDocumentName_.</p> <p>Saves it as localDocumentName_ using the loginDetails_ to pull the file from TIBCO MFT server.</p> <p>Parameters:</p> <ol style="list-style-type: none"> <li>1. remoteDocumentName_ – The remote file to pull as a String</li> <li>2. localDocumentName_ – The local file name to save as a String</li> <li>3. loginDetails_ - the LoginDetails object with the following properties populated: <ul style="list-style-type: none"> <li>• encryptedUID – Encrypted User Id</li> <li>• encryptedPWD – Encrypted Password.</li> <li>• channelSftp – SFTP Channel connected</li> </ul> </li> </ol>

#	Method	Description
3.	<pre>public String extractDocument(File localDocumentName_, File localTargetDirectory_) throws Exception;</pre>	<p>Extracts the zip file downloaded from the TIBCO MFT server.</p> <p>Parameters:</p> <ol style="list-style-type: none"> <li>1. localDocumentName_ - The local zip file to extract</li> <li>2. localTargetDirectory_ - The target directory to place the extracted contents</li> </ol> <p>Returns: The extracted Directory name as a String.</p>
4.	<pre>public boolean processMedicalDocumentation(String remoteDocumentName_);</pre>	<p>This method does the following:</p> <ol style="list-style-type: none"> <li>1. Extracts the zip file into the "download" directory using the extractDocument() method.</li> <li>2. If extraction fails, calls the acknowledge method with an error event.</li> <li>3. After successful extraction, verifies the extracted payloads against the checksum in the metadata file using the checkPayloads() method.</li> <li>4. If checksum fails, calls the acknowledge method with an error event.</li> <li>5. If checksum passes, calls the acknowledge() method with a success event.</li> </ol> <p>Parameters:</p> <ol style="list-style-type: none"> <li>1. localDocumentPath_ - The local document name to process.</li> </ol> <p>Returns: The Boolean status of the processing for that document.</p>

#	Method	Description
5.	public String acknowledge(RCPickupNotification rcPickupNotification_) throws Exception;	<p>Generates the pickup notification for a downloaded document. If the ErrorInfo object is populated, it generates an error pickup notification. If the ErrorInfo object is null, it generates a pickup notification.</p> <p>Parameters:</p> <ol style="list-style-type: none"> <li>1. rcPickupNotification_ - The RCPickupNotification object.</li> </ol> <p>Returns: The TIBCO MFT Server ready compressed file name created in the output directory as a String.</p>
6.	public boolean checkPayloads(File localExtractedDirectory_, RetrieveMedicalDocumentationResponse retrieveMedicalDocumentationResponse_ );	<p>Checks the payload files against the metadata from the package.</p> <p>Parameters:</p> <ol style="list-style-type: none"> <li>1. localExtractedDirectory_ – The directory in which the payloads were extracted to as a File.</li> <li>2. retrieveMedicalDocumentationResponse_ – the metadata xml as object.</li> </ol> <p>Returns: The status of the checksum verification.</p>

### 11.2.3 Outbound

Table 22: Retrieval of Outbound Documents Details lists the methods and their descriptions used to retrieve a list of outbound documents.

Table 22: Retrieval of Outbound Documents Details

#	Methods	Description
1.	public List<String> getOutboundDocuments(String localOutputDirectoryPath_, String localOutboundDocumentNamePattern_) throws Exception;	<p>This method is used to retrieve the list of outbound documents in the "output" directory to push.</p> <p>Parameters:</p> <ol style="list-style-type: none"> <li>1. localOutputDirectoryPath_ – The local "output" directory to push files from as a String.</li> <li>2. localOutboundDocumentNamePattern_ – The file name pattern to push as a String.</li> </ol> <p>Returns: The List&lt;String&gt; with the names of the Outbound files in the "output" directory.</p>

#	Methods	Description
2.	public void pushDocument(String localOutboundDocumentPath_, String remoteOutboundDirectoryName_, LoginDetails loginDetails_) throws Exception;	<p>This method is used to push a local compressed document from the "output" directory to the TIBCO MFT server.</p> <p>Parameters:</p> <ol style="list-style-type: none"> <li>1. localOutboundDocumentPath_ – The name of the file to push as a String.</li> <li>2. remoteOutboundDirectoryName_ – The remote directory name to push to as a String.</li> <li>3. loginDetails_ - The LoginDetails object with the following properties populated: <ul style="list-style-type: none"> <li>• encryptedUID – Encrypted User Id</li> <li>• encryptedPWD – Encrypted Password.</li> <li>• channelSftp – SFTP Channel connected.</li> </ul> </li> </ol>

#### 11.2.4 Utilities – PMDPA Result

Table 23: esMD Manual Submit PMD PA Result lists the methods and their descriptions used in handling PMD PA processing.

Table 23: esMD Manual Submit PMD PA Result

#	Methods	Description
1.	public SubmitPMDPADeterminationRequest promptForInfo() throws Exception;	<p>This method is used to prompt the user for information needed to populate the SubmitPMDPADeterminationRequest object. It uses applet prompts.</p> <p>Returns: The SubmitPMDPADeterminationRequest object populated with the data provided by the user.</p>

#	Methods	Description
2.	<pre>public String createCompressedTIBCOFileForPMPDP ARequest(SubmitPMDPADeterminationR equest submitPMDPADeterminationRequest_);</pre>	<p>This method is used create the XML file and compress it into a TIBCO MFT Server file.</p> <p>Parameters:</p> <ol style="list-style-type: none"> <li>1. submitPMDPADeterminationRequest – The SubmitPMDPADeterminationRequest object to use.</li> </ol> <p>Returns: The compressed outbound file name ready to be pushed by the outbound process.</p>

## 11.2.5 Utilities - Encryption

Table 24: Encryption details the encryption utility.

Table 24: Encryption

#	Methods	Description
1.	<pre>public String encryptKSPassword(String keyStorePassword_) throws Exception;</pre>	<p>This method encrypts the Keystore password so it can be stored in the configuration file.</p> <p>Parameters:</p> <ol style="list-style-type: none"> <li>1. keyStorePassword_ – The password to encrypt as a String.</li> </ol> <p>Returns: The Encrypted Keystore Password using “PBEWithMD5AndTripleDES”</p>
2.	<pre>public String encryptPKPassword(String privateKeyPassword_) throws Exception;</pre>	<p>This method encrypts the Private Key password so it can be stored in the configuration file.</p> <p>Parameters:</p> <ol style="list-style-type: none"> <li>1. privateKeyPassword_ – The password to encrypt as a String.</li> </ol> <p>Returns: The Encrypted Private Key Password using “PBEWithMD5AndTripleDES”</p>

#	Methods	Description
3.	<pre>public Map&lt;String, String&gt; encryptCredentials(Map&lt;String, String&gt; loginInfo_, ESMDCConfig.KeyStoreInfo keyStoreInfo_) throws Exception;</pre>	<p>This method encrypts the IACS login credentials using a RSA Public Key from the JKS Store.</p> <p>Parameters:</p> <ol style="list-style-type: none"> <li>1. loginInfo_ - The Map&lt;String, String&gt; containing the UID and PWD as keys.</li> <li>2. keyStoreInfo_ - The ESMDCConfig.KeyStoreInfo object with the following details populated: <ul style="list-style-type: none"> <li>• keyStoreLocation – The JKS Store to use as a String.</li> <li>• encKeyInfo - The Encrypted Keystore password to load the JKS as a String.</li> <li>• certAlias – The alias of the certificate to retrieve the public key as a String.</li> </ul> </li> </ol> <p>Returns: The Map&lt;String, String&gt; of encrypted login credentials ENC_UID and ENC_PWD as keys.</p>

#	Methods	Description
4.	<pre>public Map&lt;String, String&gt; decryptCredentials(Map&lt;String, String&gt; encryptedLoginInfo_, ESMDCConfig.KeyStoreInfo keyStoreInfo_) throws Exception;</pre>	<p>This method decrypts the IACS login credentials using a RSA Private Key from the JKS Store.</p> <p>Parameters:</p> <ol style="list-style-type: none"> <li>1. encryptedLoginInfo_ - The Map&lt;String, String&gt; of encrypted login credentials ENC_UID and ENC_PWD as keys.</li> <li>2. keyStoreInfo_ - The ESMDCConfig.KeyStoreInfo object with the following details populated: <ul style="list-style-type: none"> <li>• keyStoreLocation – The JKS Store to use as a String.</li> <li>• encKeyInfo - The Encrypted Keystore password to load the JKS as a String.</li> <li>• certAlias – The alias of the certificate to retrieve the public key as a String.</li> <li>• encKeyInfoExt – The Encrypted private key password to load the private key from the JKS Store as a String.</li> </ul> </li> </ol> <p>Returns: The Map&lt;String, String&gt; containing the UID and PWD as keys.</p>

### 11.2.6 Utilities - Handshake

Refer to Table 25: Remote Troubleshooting for details on the ExecuteHandshake method.

Table 25: Remote Troubleshooting

Methods	Description
<pre>public bool ExecuteHandshake()</pre>	<p>This sample method invokes a call to the TIBCO MFT server to pass login information to assist in remote troubleshooting.</p> <p>Returns: TRUE if handshake succeeded.</p>

## 11.3 Logs

Table 26: RC Client Logs lists the logs the RC Client provides. The RC is advised to monitor the logs for errors and exceptions.

Table 26: RC Client Logs

Log	Description
handshake.log	Logging for the handshake.bat utility
rc.log	Logging for the sample application.
inbound.log	Logging for the Inbound Process
outbound.log	Logging for the Outbound Process
pmdpa.log	Logging for the PMD PA File Creation utility (i.e., pmdpa.bat)

## 11.4 Utilities

Table 27: RC Client Utilities lists the utilities the RC Client provides.

Table 27: RC Client Utilities

Utility	Description
handshake.bat	Allows the user to check the connection to TIBCO MFT Server. Also allows esMD team to debug any connection issues remotely.
encryptConfig.bat	Encrypts the provided passwords and updates the configuration XML file.
rcclient.bat	The RC Client sample program.
pmdpa.bat	Allows the RC to submit PMD PA file.

## 12. Error Codes

---

Table 28: Error Codes lists the types of error codes with their descriptions. These codes are used to populate the ErrorInfo object inside the error pickup notification XML (e.g., R\_TID\_Pickup\_Error\_Request.xml).

Table 28: Error Codes

Error Type	Error Code	Description
UNZIP ERROR	534	ESMD_534 - RC Client processing error (Unzip failure). Please resubmit.
CHECKSUM ERROR	535	ESMD_535 - RC Client processing error (Checksum issue). Please resubmit.
METADATA ERROR	536	ESMD_536 - RC Client processing error (Metadata issue). Please resubmit.

## 13. Contact Information

---

Table 29: Support Points of Contact lists the Points of Contact (POC) for esMD.

Table 29: Support Points of Contact

Contact	Phone	Email	Hours of Operation
CMS esMD Help Desk	(443) 832-1856	<a href="mailto:CMSesMDHelpdesk@gssinc.com">CMSesMDHelpdesk@gssinc.com</a>	Normal Business Days 8 AM to 8 PM ET

## Appendix A: New User Registration

This appendix provides instructions for creating a new IACS user account.

1. To register in IACS, access the CMS website. Copy and paste the following link in your browser, or press Ctrl and click the link:  
<https://idm.cms.hhs.gov/idm/user/welcome.jsp>

Figure 7: New User Registration shows the Request Access window that opens.

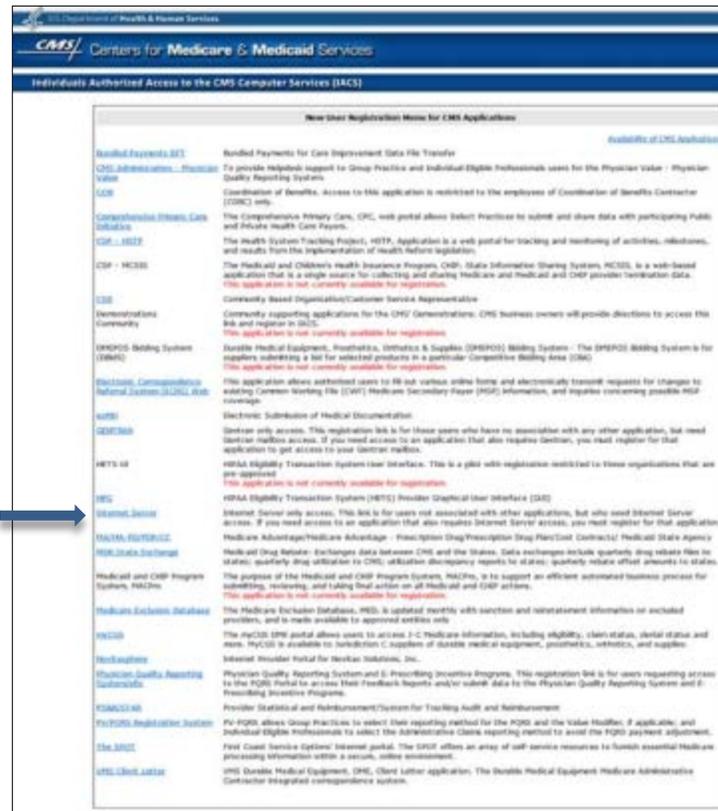
Figure 7: New User Registration



2. Click the **New User Registration** hyperlink (also indicated with an arrow in Figure 7: New User Registration).

Figure 8: New User Registration Menu shows the next window that opens.

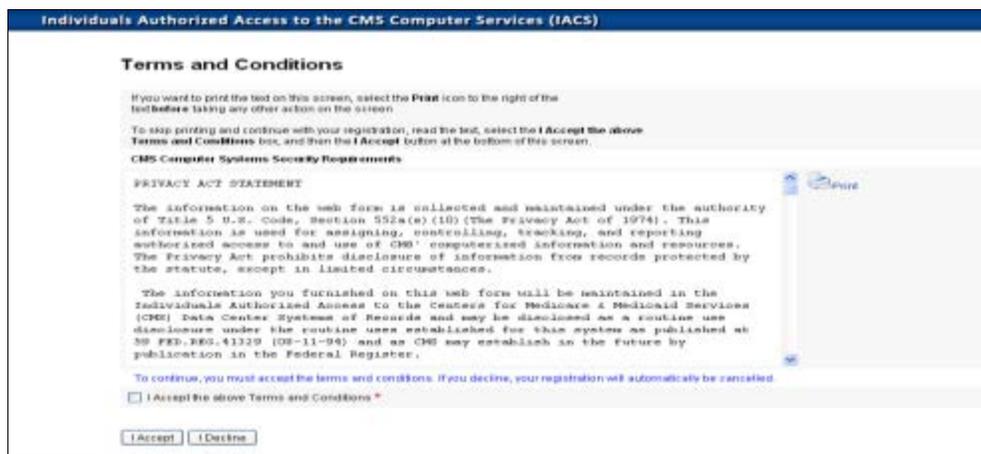
Figure 8: New User Registration Menu



3. Click the **Internet Server** hyperlink (also indicated by an arrow in Figure 8: New User Registration Menu).

Figure 9: Terms and Conditions shows the Privacy Act Statement and the Rules of Behavior, which are the required terms and conditions for accessing CMS computer systems.

Figure 9: Terms and Conditions



4. After you read the content and agree, click the **I Accept** button.

Figure 10: New Registration Form shows the New User Registration page selected in the form.

Figure 10: New Registration Form

The screenshot shows the 'New User Registration' form within the CMS IACS interface. The form is titled 'New User Registration' and is part of the 'Individuals Authorized Access to the CMS Computer Services (IACS)' system. The form includes a progress bar with steps: 'New User Registration' (selected), 'Email Verification', 'Contact Information', 'Authentication Questions', 'Review Request', and 'Acknowledgement'. Below the progress bar, there is a message: 'CMS is authorized to validate your personal information using your legal name, Date of Birth and Social Security Number.' followed by a button 'Enter Random Test-data'. The form fields are: 'Title' (dropdown), 'First Name' (text field with asterisk), 'Last Name' (text field with asterisk), 'Suffix' (dropdown), 'Middle Initial' (text field), 'Professional Credentials' (text field with example: 'Example: MD, RN, LPN, MBA, PhD, etc. (Limit 12 characters)'), 'Social Security Number' (text field with asterisk and format: 'Valid SSN Format is XXX-XX-XXXX'), 'Date of Birth' (text field with asterisk and format: 'Valid Date of Birth format is mm/dd/yyyy'), 'E-mail' (text field with asterisk and format: 'Valid E-mail address format is user@nternetprovider.domain. List of allowed domains: \\.com, .gov, .net, .org, .us, .mil, .biz, .edu, .pro'), and 'Confirm E-mail' (text field with asterisk). At the bottom left, there are 'Next' and 'Cancel' buttons. The footer of the form displays 'OMB: 0938-0989'.

5. Complete the following required fields indicated by an asterisk (\*) and the optional fields if desired.
  - The First Name and Last Name
  - Social Security Number (SSN) in the format specified
  - Date of birth in the format specified
  - A unique, work-related email address where you can be reached.
  - The same email address for confirmation.
6. Click **Next** and keep the next window open.

**Note:** If you close the window, the entire process will be lost.

**Note:** If you click the Cancel button during this registration process, the request will be cancelled and all information that was entered will be lost. The system displays a warning message about your cancellation. Clicking OK confirms that you want to cancel. Clicking Cancel stops the cancellation so that you can continue the registration. If you cancel, you must click OK a second time to close the browser.

If you clicked **Next**, the system validates your SSN and email address to verify they do not already exist for another IACS account.

If your information is successfully validated, the system sends you an email (at the email address you provided). The Subject line will read "IACS: E-mail Address Verification". The email will contain an eight-digit verification code.

- Type this code accurately (without spaces or characters) in the Verification Code field shown in Figure 11: Email Verification. The verification code will not work properly if cut or copied and pasted it into the field.

**Note:** You have up to 30 minutes to access your email, obtain the verification code, and enter it on this page.

Figure 11: Email Verification

If you do not receive the verification email, click the **Re-send verification code** link to the right of the Verification Code field.

You can request a resend up to three times or contact the Help Desk. If you realize you may have entered an incorrect email address, you must start the entire process again. After three unsuccessful email attempts, you must start the entire process again.

- After you successfully enter the verification code, click **Next**.

The Contact Information page is highlighted, as shown in Figure 12: New Registration - Contact Information, and now contains pre-populated fields.

Figure 12: New Registration - Contact Information

- As on the previous pages, fill in the required fields indicated by an asterisk (\*) and click **Next**.

After the data is validated, the system displays the **Authentication Questions** page, shown in Figure 13: Authentication Questions.

Figure 13: Authentication Questions

Question	Answer
What is your grandmother's maiden name?	esmd
What was the model of your first car?	esmd
What is the middle name of your oldest cousin?	esmd
What was the name of your first pet?	
What was your childhood phone number?	
What was the first name of your first boyfriend?	
What was the first name of your first girlfriend?	
What is the name of your first elementary school?	
What was your childhood street name?	
What was the name of your first employer?	
What was your grandfather's profession?	
What was the name of your first college roommate?	
Where was your wedding reception held?	

- Answer a minimum of two Authentication Questions and click **Next** to complete your registration.

These answers will be used to validate your identity if you need to recover your user ID or password using IACS' **Forgot your User ID?** or **Forgot your Password?** self-service features.

The system displays the **Review Registration Details** page, shown in Figure 14: Review Registration Details.

Figure 14: Review Registration Details

**Review Registration Details**

The following is the information you entered on the New User Registration Form. Please review the information below to verify correctness.

- To modify any of the information, click **Edit**.
- If the information is correct and you wish to proceed, click **Submit**.

<b>First Name:</b> John Doe	<b>MI:</b> N	<b>Last Name:</b> Admin
<b>Title:</b> Mr.	<b>Suffix:</b> Jr.	<b>Professional Credentials:</b> BA
<b>Social Security Number:</b> *****9999		
<b>Date of Birth:</b> 10/18/1972		
<b>E-mail:</b> pthompson-2012_02-4570@idm.com		
<b>Office Telephone:</b> 456-456-4522 X452		
<b>Company Name:</b> ABCD Inc.	<b>Company Telephone:</b> 451-452-4529 X453	
<b>Address 1:</b> 101 Main Street	<b>Address 2:</b> Suite 102	
<b>City:</b> Baltimore	<b>State/Territory:</b> MD	<b>Zip Code:</b> 45274-4535
<b>User Type:</b> esMD		
<b>Role:</b> esMD Admin		

Question	Answer
What is your grandmother's maiden name?	esmd
What was the model of your first car?	esmd
What is the middle name of your oldest cousin?	esmd

11. Review your information.

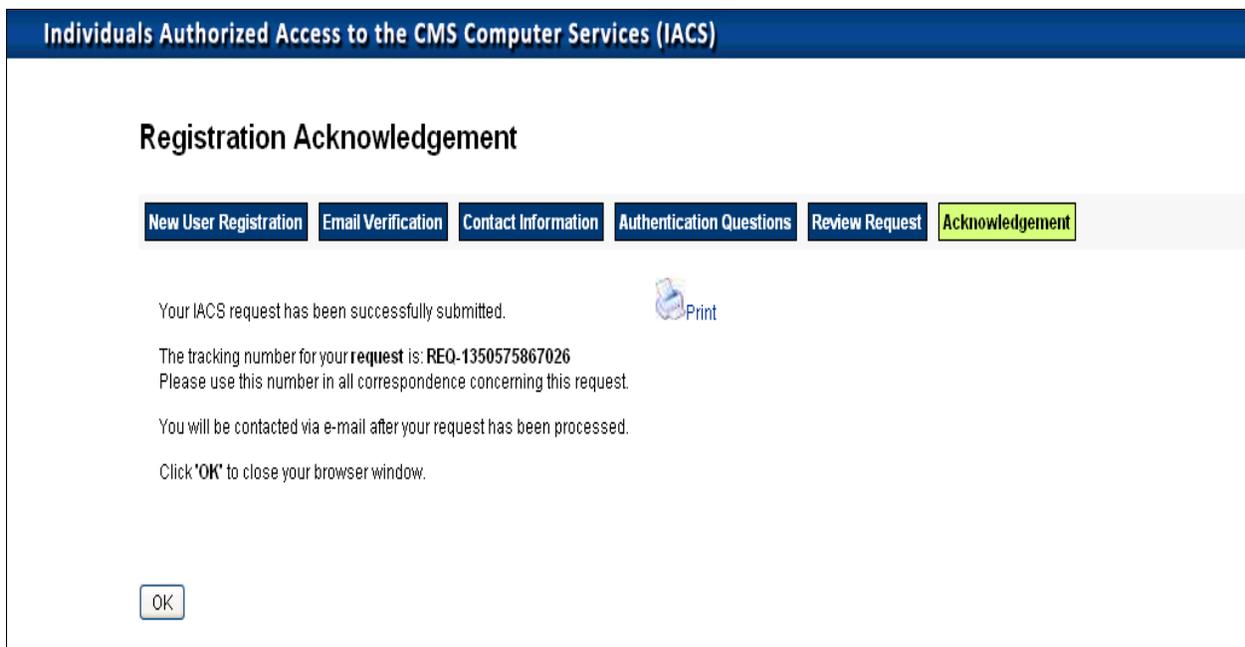
**Note:** If you need to change anything, click the **Edit** button to make the change. The New User Registration pages will be displayed to make any changes. However, you cannot change the E-mail and Confirm E-mail fields using the **Edit** button.

12. Click the **Submit** button when you are satisfied that your registration information is correct.

The Acknowledgement is displayed, as shown in Figure 15: Registration Acknowledgement, indicating your registration request was successfully submitted and a request tracking number was assigned. Record and use this number when you have questions about the status of your request. You will be contacted by email within 48 hours.

**Note:** You can print your registration information by clicking the **Print** icon.

Figure 15: Registration Acknowledgement



13. Click **OK** (this is required to complete your registration).

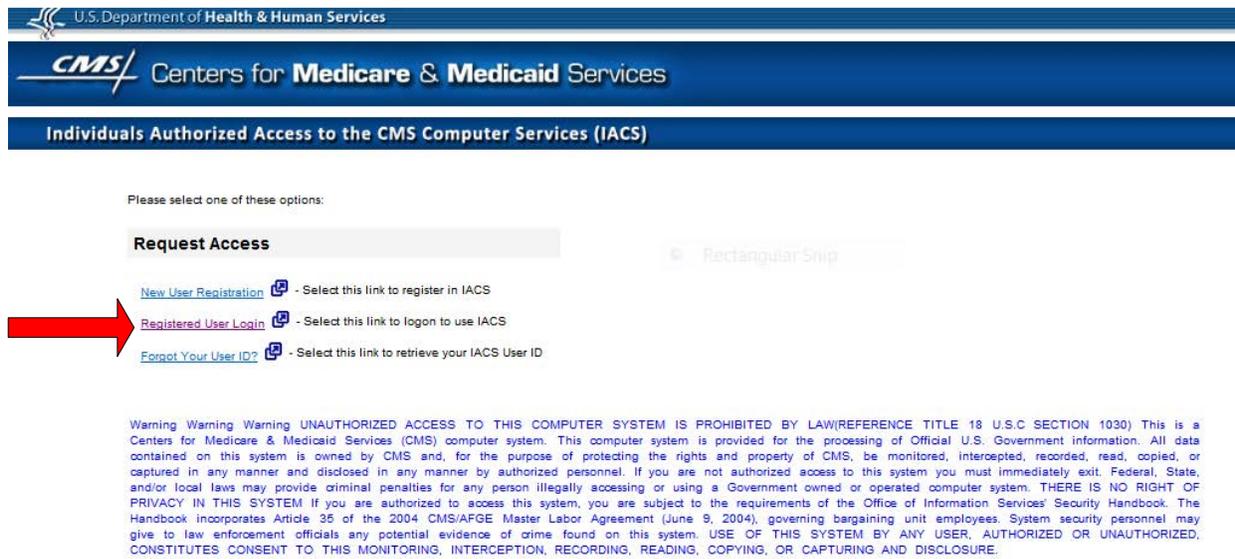
# Appendix B: Registered User Login Instructions

This appendix provides instructions for existing IACS account users.

1. To login into IACS, access the CMS website. Copy and paste the link in your browser, or press Ctrl and click the link: <https://idm.cms.hhs.gov/idm/user/welcome.jsp>

Figure 16: Registered User Login Window shows the Request Access window that opens.

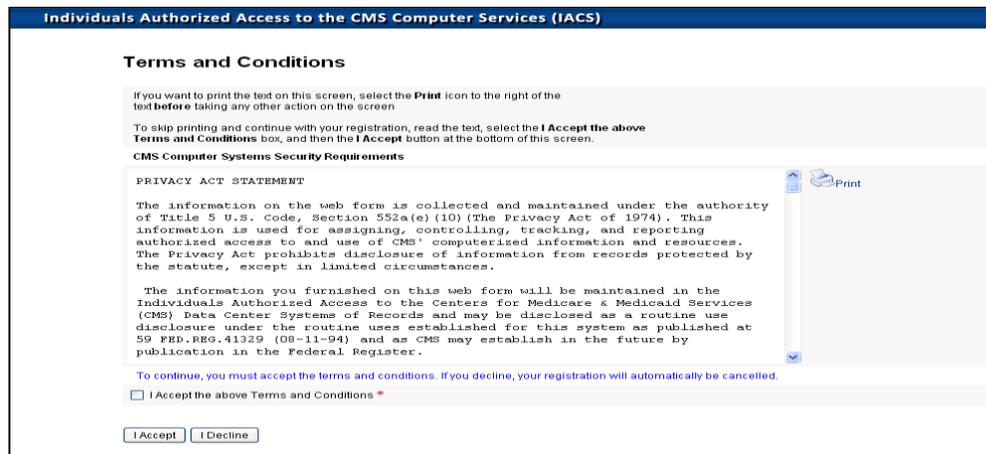
Figure 16: Registered User Login Window



2. Click the **Registered User Login** hyperlink (also indicated by the arrow in the image above).

Figure 17: Terms and Conditions shows the Privacy Act Statement and the Rules of Behavior, which are the required terms and conditions for accessing CMS computer systems.

Figure 17: Terms and Conditions



3. After you read the content and agree, click the **I Accept** button.

Figure 18: My Profile is the next window that opens.

Figure 18: My Profile



4. Click **Modify Account Profile** (also indicated by a red arrow in Figure 18: My Profile).

Figure 19: **Modify Account Profile** is the next window that opens and the fields will be populated with the user's information.

**Note:** The fields in the example are empty for security purposes.

Figure 19: Modify Account Profile

U.S. Department of Health & Human Services  
**CMS** Centers for Medicare & Medicaid Services  
 Individuals Authorized Access to the CMS Computer Services (IACS)

### Modify Account Profile

**Modify Account Profile** | Email Verification | Review Request | Acknowledgement

**User Information**

User ID:

Title:  First Name:  Last Name:  Suffix:

Middle Initial:  Professional Credentials:

Date of Birth:  Valid Date of Birth format is mm/dd/yyyy

E-mail:

Valid E-mail address format is user@internetprovider.domain. List of allowed domains: com, gov, net, org, us, mil, biz, edu, vl, or, md, coop

**Professional Contact Information**

Office Telephone:  Ext.:  Valid Telephone Number Format is XXX-XXX-XXXX

Company Name:  Company Telephone:  Ext.:

Country:

Address 1:  Address 2:

City:  State/Territory:  Zip Code:

**Access Request**

Select Action:

Select Application:  Availability of CMS Applications

Justification for Action:

5. Verify user information is correct within the populated fields.
6. Under Access Request, in the Select Action field, select and click Add Application.
7. The same screen will refresh.
8. Select and click Select Application.
9. Click the Internet Server application.
10. Enter appropriate information in the Justification for Action field.

## Appendix C: XML Message Details

Table 30: XML Message Details provides details of the XML messages transferred between the esMD application and the RC Client.

**Note:** TID stands for the Transaction ID.

Table 30: XML Message Details

Business	#	Folder	File Name	Process
PMDPA	1	P_TID	P_TID_Pickup_Request.xml	Outbound
	2	R_TID	R_TID_Pickup_Virus_Scan_Error_Response.xml	Inbound
	3	N_TID	N_TID_Pickup_HIH_Status_Response.xml	Inbound
	4	R_TID	R_TID_Pickup_Error_Request.xml	Outbound
	5	E_TID	E_TID_PMDPA_Review_Result_Request.xml	Outbound
	6	R_TID	R_TID_PMDPA_Review_Result_Virus_Scan_Error_Response.xml	Inbound
	7	R_TID	R_TID_PMDPA_Review_Result_Validation_Error_Response.xml	Inbound
	8	N_TID	N_TID_PMDPA_Review_Result_HIH_Status_Response.xml	Inbound
ADR	9	P_TID	P_TID_Pickup_Request.xml	Outbound
	10	R_TID	R_TID_Pickup_Virus_Scan_Error_Response.xml	Inbound
	11	N_TID	N_TID_Pickup_HIH_Status_Response.xml	Inbound
	12	R_TID	R_TID_Pickup_Error_Request.xml	Outbound
ADMC	13	P_TID	P_TID_Pickup_Request.xml	Outbound
	14	R_TID	R_TID_Pickup_Virus_Scan_Error_Response.xml	Inbound
	15	E_TID	N_TID_Pickup_HIH_Status_Response.xml	Inbound
	16	R_TID	R_TID_Pickup_Error_Request.xml	Outbound
APPEAL	17	P_TID	P_TID_Pickup_Request.xml	Outbound
	18	R_TID	R_TID_Pickup_Virus_Scan_Error_Response.xml	Inbound
	19	N_TID	N_TID_Pickup_HIH_Status_Response.xml	Inbound
	20	R_TID	R_TID_Pickup_Error_Request.xml	Outbound
RAC Request	21	P_TID	P_TID_Pickup_Request.xml	Outbound
	22	R_TID	R_TID_Pickup_Virus_Scan_Error_Response.xml	Inbound
	23	R_TID	R_TID_Pickup_HIH_Status_Response.xml	Inbound
	24	R_TID	R_TID_Pickup_Error_Request.xml	Outbound
*PMDPA Non- Emergent & Hyper-baric PA Requests	25	P_TID	P_TID_Pickup_Request.xml	Outbound
	26	R_TID	R_TID_Pickup_Virus_Scan_Error_Response.xml	Inbound
	27	N_TID	N_TID_Pickup_HIH_Status_Response.xml	Inbound
	28	R_TID	R_TID_Pickup_Error_Request.xml	Outbound

**Note:** With Release 3.1, esMD will support Non-Emergent Ambulance Transport and HBO PA Requests. These requests will come in as PMD PA Requests with a Content Type Code of "8" to selected AB MACs.

## Acronyms

Table 31: Acronyms

Acronym	Literal Translation
<b>ADMC</b>	Advance Determination of Medicare Coverage
<b>ADR</b>	Additional Documentation Request
<b>API</b>	Application Programming Interface
<b>BDC</b>	CMS Baltimore Data Center
<b>CMS</b>	Centers for Medicare & Medicaid Services
<b>DME</b>	Durable Medical Equipment
<b>ECM</b>	Enterprise Content Management
<b>EFT</b>	Enterprise File Transfer system
<b>esMD</b>	Electronic Submission of Medical Documentation
<b>ET</b>	Eastern Time
<b>HBO</b>	Hyperbaric Oxygen
<b>HHS</b>	Health and Human Services
<b>HIH</b>	Health Information Handler
<b>IACS</b>	Individuals Authorized Access to CMS Computer Service
<b>ID</b>	Identification
<b>Javadoc</b>	Java documentation
<b>JCE</b>	Java Cryptography Extension
<b>JDK</b>	Java Development Kit
<b>JKS</b>	Java Key Store
<b>JRE</b>	Java Runtime Environment
<b>MB</b>	Megabytes
<b>MFT</b>	Managed File Transfer
<b>NPI</b>	National Provider Identifier
<b>OID</b>	Object Identifier
<b>PA</b>	Prior Authorization
<b>PMD</b>	Power Mobility Devices
<b>RAC</b>	Recovery Audit Contractor
<b>RC</b>	Review Contractor
<b>RSA</b>	Rivest, Shamir & Adleman (public key encryption technology)
<b>SFTP</b>	SSH File Transfer Protocol
<b>SSH</b>	Secure Shell
<b>SSN</b>	Social Security Number
<b>TID</b>	Transaction ID
<b>XLC</b>	Expedited Life Cycle
<b>XML</b>	Extensible Markup Language

## Glossary

Table 32: Glossary

Term	Definition
<b>Application Programming Interface</b>	In computer programming, an Application Programming Interface (API) specifies how some software components should interact with each other and can be used to ease the work of programming graphical user interface components.
<b>Centers for Medicare &amp; Medicaid Services</b>	The Centers for Medicare & Medicaid Services (CMS) is the Department of Health and Human Services (HHS) agency responsible for Medicare and parts of Medicaid.
<b>Electronic Submission of Medical Documentation</b>	Electronic Submission of Medical Documentation (esMD) is a mechanism for submitting medical documentation via an internet gateway connecting Providers to the CMS. In its second phase, esMD will enable Medicare Review Contractors (RCs) to electronically send claim related Additional Document Request (ADR) letters to Providers when their claims are selected for review.
<b>Extensible Markup Language</b>	Extensible Markup Language (XML) is a set of rules for encoding documents electronically. It is defined in the XML 1.0 Specification produced by the World Wide Web Consortium (W3C) and several other related specifications; all are fee-free open standards.
<b>Health Information Handler</b>	A Health Information Handler (HIH) is any company that handles information on behalf of a provider or Durable Medical Equipment (DME) supplier. Examples include Release of Information vendors, Health Information Exchanges, Regional Health Information Organizations, Electronic Health Record vendors, and Claim Clearinghouses.
<b>Javadoc</b>	Java documentation generator automatically generates documentation from source code comments.
<b>Java Cryptography Extension</b>	Java Cryptography Extension (JCE) is an Application Program Interface (API) that provides a uniform framework for the implementation of security features.
<b>Java Development Kit</b>	The Java Development Kit (JDK) is a software development environment for writing applets and application in Java.
<b>Java Runtime Environment</b>	Java Runtime Environment (JRE), also known as Java Runtime, is part of the Java Development Kit (JDK), a set of programming tools for developing Java applications.
<b>National Provider Identifier</b>	The National Provider Identifier (NPI) is a unique identification number for use in standard health care transactions. The NPI is issued to health care providers and covered entities that transmit standard Health Insurance Portability & Accountability Act of 1996 (HIPAA) electronic transactions (e.g., electronic claims and claim status inquiries).
<b>Object Identifier</b>	Object Identifier (OID) is an identifier used to name an object. In computer security, OIDs serve to name almost every object type in X.509 certificates, such as components of Distinguished Names.
<b>PMD PA</b>	Power Mobility Devices Prior Authorization (PMD PA) is a covered item of Durable Medical Equipment (DME) that is in a class of wheelchairs that includes power wheelchairs.

<b>Term</b>	<b>Definition</b>
<b>Review Contractor</b>	A Review Contractor (RC) is an entity designated as a recipient of requested medical documentation. Examples are Recovery Audit Contractors (RACs), Medicare Administrative Contractors (MACs), DMACs, and Payment Error Rate Measurement (PERM) or Comprehensive Error Rate Testing (CERT) contractors.
<b>Social Security Number</b>	A Social Security Number (SSN) is a unique identification number assigned to individuals by the Social Security Administration (SSA).

## Record of Changes

**Table 33: Record of Changes**

Version Number	Date	Author/Owner	Description of Change
1.2	08/28/2014	Melony Stehlik, Jim Runser	Initial document to identify the changes implemented with Release 3.1 despite no changes being made to the RCs' code. Updated Help Desk hours.
1.3	09/11/2014	Melony Stehlik	Updated documented per CMS comments to the .NET version of the RC Client Implementation Plan.  Adjusted version number; updated Table 22 with New Lines of Business details.
1.4	09/23/2014	Jim Runser	Reversioned to maintain continuity throughout Release 3.X series.
1.5	09/30/2014	Faye Newsham	Updated for Section 508 compliance.

## Approvals

The undersigned acknowledge that they have reviewed the RC Client Implementation Guide and agree with the information presented within this document. Changes to this Document will be coordinated with, and approved by, the undersigned, or their designated representatives.

Signature: \_\_\_\_\_ Date: \_\_\_\_\_

Print Name: Braeyon Terry-Connor

Title: Contracting Officer's Representative

Role: CMS Approving Authority