Centers for Medicare & Medicaid Services

Electronic Submission of Medical Documentation (esMD)

Review Contractor (RC) Client Java User Guide and Installation Handbook AR2025.04.0

Version 18.1

01/13/2025

Document Number: AR2025.04.0_esMD_RC_Implementation_Guide_Java
Contract Number: HHSM-500-2016-00068I:75FCMC21F0001

Document Change History

Table 1: Record of Changes

Version Number	Date	Author/Owner	Description of Change
18.1	01/07/2025	Venkata Gurram	Changes for April 2025 release version 18.1 include: 1. Added new Appendix: E. 2. Added new Table:
18.0	01/06/2025	Venkata Gurram	Changes for April 2025 release version 18.0 include: 3. Added new Table: 11. 4. Updated Figure: 55. 5. Updated Sections: 1.4, 11.2.3, and 11.2.17
17.2	08/27/2024	Venkata Gurram	Changes for October 2024 release version 17.2 include: 6. Updated Table: 45.
17.1	07/26/2024	Srilakshmi Akula	Changes for October 2024 release version 17.1 include: 7. Updated Tables: 42,43,45, and 49. 8. Updated Section: 26.2.1 - Added a note.
17.0	07/15/2024	Venkata Gurram	Changes for October 2024 release version 17.0 include: 1. Relabeled Figures 69 and 75 to Tables 13 and 14, then adjusted the Caption numbering for all subsequent figures and tables. 2. Added Section: 26.2.3. 3. Added Table: 45. 4. Updated Sections: 1.4, 1.4.1, 1.4.2, 10, 11.1, 11.1.1, 11.1.16, 11.2, 11.2.10, 11.2.11, 11.2.17, 19, 23.2.1.1, 23.2.2, 23.3, and 26.2. 5. Updated Figures: 19, 35, 36, 37, 38, 42, 49, 50, 55, and 94. 6. Updated Tables: 9, 23, 27, 42, 46, 51, and 52.
16.1	05/10/2024	Venkata Gurram	Changes for July 2024 release version 16.1 include: 1. Updated the following Figures: a. 22, 29, 30, 45, 46, 49, and 50: Updated to display an updated SHA-256 checksum value. 2. Updated the following Sections: a. 23.6.4: Added note for the checksum. 3. Updated the following Tables: a. 23: Updated row 12 with a note for the checksum. b. 50: Added acronym SHA.

Version Number	Date	Author/Owner	Description of Change
16.0	04/24/2024	Venkata Gurram	Changes for July 2024 release version 15.2 include: 1. Updated the following Sections: a. 1.4.3: Updated password policy information. b. 8.1.1: Updated RC Client download link. 2. Updated the following Tables: a. 10: Added row 9 for Corrupted PDF file.
15.1	03/22/2024	Venkata Gurram	Changes for April 2024 release version 15.1 include: 1. Updated Section 11.2.10 with PDF filename format. 2. Updated Section 23.4.1 with successful login details. 3. Updated Table 42 with PDF filename format.
15.0	01/22/2024	Venkata Gurram	Changes for April 2024 release version 15.0 include: 1. Converted Section 13 subsections into individual sections to improve organization. 2. Applied "Figures" caption to all images which lacked captions and updated the List of Figures. 3. Converted the "Tables" caption to a "Figures" caption for all transaction data samples and updated the List of Figures. 4. Updated Sections: 1.4, 1.4.1, 8.1.1, 11.2.10, and 17. 5. Updated Figure 49. 6. Updated Tables 36, 39, 42, 47, 48, and 49.
14.1	08/03/2023	Venkata Gurram	Changes for the October 2023 release Version 14.1 include: 1. Updated Section 8.1.1 to reflect new Java RC Client software download link..
14.0	08/01/2023	Venkata Gurram	Changes for the October 2023 release include: 1. Updated Sections: 1.4, 1.4.1, 1.4.2, 1.4.3, 8.1.1, 10, 11.1, 11.1.4, 11.1.13, 11.1.16, 11.2, 11.2.17, 12.1, 12.6, 13.5, 13.6, 14.3, 15.2.4, and 18.1.2.1. 2. Updated Tables: 10, 11, 23,24, 26, 27, 28, 46, 52, 56, 70, 71, 73, 78, and 80. 3. Updated Figures: 5, 7, 24, 27 and 28.
13.0	03/31/2023	Venkata Gurram	Changes for the July 2023 release include:

Version Number	Date	Author/Owner	Description of Change
			1. Updated Sections: 1.4, 1.4.2, 3, 4, 5, 8.1.4, 10, 11.1, 11.1.15, 11.2, 11.2.1, 11.2.2, 11.2.3, 11.2.4, 11.2.9, 11.2.15, 12.1, 12.1.9, 12.1.10, 12.1.14, 2.1.18, 12.2, 12.2.11, 12.2.13, 12.2.14, 13.1, 13.2, 13.3.2, 13.4, and 14.1. 2. Updated Tables: 9, 10, 11, 12, 14, 16, 17, 19, 20, 23, 25, 27, 28, 29,30, 31, 32, 33, 34, 35, 38, 39, 41, 42, 43, 44, 45, 49, 62, 64, 68, 70, 75, and 76. 3. Updated Figures: 7, 18, and 22.
12.0	09/16/2022	Venkata Gurram	Changes for the January 2023 release include: 1. Sections 1.4.2, 1.5, 1.5.1, 8, 8.2, 12.1.11, 12.1.12, 12.1.13, 12.1.14, 12.1.15, 12.1.16, 12.1.17, 12.2.5, 12.2.6, 12.2.7, 12.2.8, 12.2.9, 12.2.12, 14.2, 14.3, 14.4, 14.5, 16.2.2 2. Changes in Tables 3, 9, 11, 21, 22, 23, 24, 25, 26, 34, 35, 36, 37, 38, 41, 45, 48, 49, 70 3. Changes in Figures 23 4. Replaced all 'ZPIC' references with 'UPIC' (Unified Program Integrity Contractor). 5. Removed RA (Recovery Auditor) from Appendix E: Acronyms 6. Updated ZPIC to UPIC (Unified Program Integrity Contractor) to Appendix E: Acronyms 7. Updated Appendix H: Approval to reflect new COR.
11.0	08/16/2022	Venkata Gurram	Changes for the January 2023 release include: 1. Sections 1.2, 1.4, 1.4.1, 1.4.2, 1.4.3, 2, 3, 4, 5, 6, 7, 7.7, 8.2, 8.1.2, 9, 12.1.14, 12.1.16, 12.2.1, 12.2.2, 12.2.8,12.2.9, 12.2.14, 13.1, 13.2, 14.1, 14.2, 14.3, 14.3.1, 14.3.2, 14.4, 14.5, 14.8, 15.3.1, 15.4, 15.4.1, 15.4.2, 15.4.3, 15.4.4, 15.5, 15.5.1, 15.5.3, 15.6.1, 15.6.2, 15.7, 15.8, 15.9, 16.2, 16.2.2, 16.2.3, 16.2.4. 2. Tables 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 42, 43, 44, 45, 47, 50, 51, 52, 53, 54, 56, 57, 67, 69, 73, 76. 3. Figures 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26.
10.0	01/27/2022	Khader Mohammad/Lisa Massengill	Changes for the April 2022 release include: 1. Section 1.4, 4, 11.1, 18.1.

Version Number	Date	Author/Owner	Description of Change
			2. Changes in Figure 7 and Tables 31 and 66. 3. Updated Section 4, steps 2, 3, 4, and 5.
9.0	10/21/2021	Khader Mohammad/Lisa Massengill	Changes for the January 2022 release include: 1. Sections 1.4, 1.4.1, 12.1,17.1 and 12.2. Changes in tables 66, 80, 85, and 86. 2. Removed all references to RRL and PA/PCR in Tables 26, 27, 28, 29, 30, 53, 54, 72, and 73. 3. Removed Figures 11,12. 4. Renumbered Tables
8.2	7/9/2021	Rohini Kolan	1. Included October release functionality in section 1.4 2. Updated sections 3, 4, 12.1.2, 12.1.23, 15.2.2
8.1	1/21/2021	Steven Prest	1. Replaced broken image in Figure 19
8.0	1/15/2021	Karthik Srinivasan	1. Updated document to reflect changes from EIDM to IDM. 2. Updated section 1.4.3 3. Updated Table 82 – Request Level UTN: Changed tracking number from “1-50 Alphanumeric Characters” to “14 Alphanumeric Characters”. 4. Replaced all “EIDM” links with “IDM” links
7.2	10/13/2020	Vijayalakshmi Muthukrishnan	1. Updated Table 78
7.1	08/11/2020	Karthik Srinivasan	1. Updated document to reflect changes in comments from TOSS
7.0	08/07/2020	Karthik Srinivasan	1. Included November release functionality in section 1.4 2. Removed all references to Coversheet and FFR 3. Updated sections 12.1, 12.1.2, 12.1.4.1, 12.1.3, 12.1.5, 12.2, 14.2, 14.3, 19.1, 19.1.2.1, 4. Updated Figure 24 5. Updated Tables 11, Table 78* (Error codes sent from esMD to RC), Table 79*, Table 81*, 84*, 88*, 89* 6. Added Appendix G - FAQs
6.1	04/20/2020	Karthik Srinivasan, Vijayalakshmi Muthukrishnan	1. Added a. Sections 12.1.26, 12.2.18, 12.2.19, 12.2.20, 14.5, 15 b. Figure 18 c. Table 10 2. Updated a. Section 1.4 – to include AR2020.07.0 release details

Version Number	Date	Author/Owner	Description of Change
			b. Section 1.4.1 – to include Document Code File Request and pickup notification for RC client functionality c. Section 12.1 & 12.2 – added Document code file d. Section e. Table 9 – Included ICDT Directory f. Table 79 – Included Document Code Flat File Validation Error g. Table 80 – Included Error Document Codes Validate File h. Table 81 i. Table 87 – Included CTC 8.5 & 17 j. Table 88 – Included CTC 17
6.0	09/30/2019	Vijayalakshmi Muthukrishnan	1. Updated for AR2020.01.0: a. Sections 1.4, 9.2, 11.1, and 11.2 b. Tables 10, 74, 77, and 83. 2. Added: a. Sections 11.1.24, 11.1.25, 11.2.15 through 11.2.16, 13.5, and 15.8 b. Figures 20 and 21. c. Tables 33, 34, 47 through 49, 54, 55, and 71.
5.1	04/09/2019	Vijayalakshmi Muthukrishnan	Resolved review comments. Updated: 1. Section 1.4 2. Table 72.
5.0	03/28/2019	Vijayalakshmi Muthukrishnan	1. Updated for AR2019.07.0: a. Sections 1.4, 1.4.1, 3 through 6, 8.2, 9, 11.1.1, 11.1.4, 11.1.7, 11.1.8, 13.8.1, 16.2.2, 17.2, and Appendix A b. Figures 10 through 13 c. Tables 12, 14,1 5, 17 through 35, 37 through 44, 55, 64, 66, 69, 70, and 72 through 75. 2. Added: a. Sections 9.1, 9.2, 10, 13.4, 15, and 18 b. Figures 7 through 9 and 19 c. Tables 10, 11, 16, 36, 46, 54, and 67.
4.1	7/16/2018	Vijayalakshmi Muthukrishnan	Resolved CMS review comments. Updated sections 1.4.1 and 1.5
4.0	06/27/2018	Vijayalakshmi Muthukrishnan	Updated for AR2018.10.0: 1. Updated: a. Sections 1.4, 11.1, and 11.2 b. Tables 11, 84, and 93 through 97. 2. Added: a. 11.1.28, 11.1.29, 11.2.22, 11.2.23, 13.5, and 14.2.18 b. Figure 17 c. Tables 39, 40, 61, 63, and 81.

Version Number	Date	Author/Owner	Description of Change
3.2	4/6/2018	Vijayalakshmi Muthukrishnan	1. Updated section 15.1 2. Added sections 11.1.26 and 11.1.27
3.1	04/02/2018	Jim Runser	Resolved CMS review comments. Updated: 1. Section 1.4 paragraph 14 2. Table 88.
3.0	03/13/2018	Vijayalakshmi Muthukrishnan	Updates for AR2018.07.0: 1. Updated: a. Sections 1.4, 1.4.1, 11.1, 11.1.5, and 11.2 b. Tables 10, 11, 78, 79, 85, and 88 through 90. 2. Added: a. Sections 11.1.25, 11.2.20, 11.2.21, 13.4, and 14.2.17 b. Tables 36, 55 through 57, and 74 c. Figure 16.
2.1	01/29/2018	Vijayalakshmi Muthukrishnan	Resolved CMS comments: 1. Updated Section 1.4 paragraph 13 2. Moved: a. Former Sections 14.1 and 14.2 to Section 14.2.9 and 14.2.10, respectively b. Former Tables 49 and 54 to Tables 62 and 63, respectively. 3. Deleted Section 14.
2.0	01/10/2018	Vijayalakshmi Muthukrishnan	Original update for AR2018.04.0: 1. Updated: a. Sections 1.4, 1.4.1, 11.1, 11.1.3, 11.1.4, 11.2, and 11.2.6 b. Tables 1, 72, and 81 through 84 c. Figures 7 through 10. 2. Added: a. Sections 11.1.1, 11.1.22, 11.1.23, 11.1.24, 11.2.18, 11.2.19, 13.3, and 15.2.14. b. Tables 33 through 35, 53, and 69 c. Figure 15.
1.0	10/04/2017	Vijayalakshmi Muthukrishnan	No substantive changes for Release AR2018.01.0 added.

Table of Contents

List of Figures

List of Tables

1. Introduction

The Centers for Medicare & Medicaid Services (CMS) is a federal agency that ensures health care coverage for more than 100 million Americans. The CMS administers Medicare and provides funds and guidance for all the 50 states in the nation, for their Medicaid programs and Children's Health Insurance Program (CHIP). The CMS works together with the CMS community and organizations in delivering improved and better coordinated care.

1.1 Overview of the esMD System

Each year, the Medicare Fee-For-Service (FFS) Program makes billions of dollars in estimated improper payments. The CMS employs several types of Review Contractors (RC) to measure, prevent, identify, and correct these improper payments. RCs find improper payments and manually review claims against medical documentation obtained to verify the providers' compliance with Medicare rules. The RCs request medical documentation by sending a paper letter to the provider. In the past, medical documentation providers had only two options for delivering the medical documentation requested by sending it by letter or fax.

The Electronic Submission of Medical Documentation (esMD) system gives providers the option of sending medical documentation electronically to a requesting RC, instead of sending the documentation by letter or fax.

Many providers use a Health Information Handler (HIH) organization to perform tasks, such as submitting claims and providing electronic health record systems. Any organization that handles health information on behalf of a provider is an HIH. Some HIHs are beginning to offer esMD gateway services; Claim Clearinghouses, Release of Information vendors, Health Information Exchanges, and Electronic Health Record vendors are often referred to as HIHs.

The esMD system allows providers and HIHs to use gateway services to send responses for requests for additional documentation electronically to an RC during the claims review process.

1.1.1 The esMD Claim Review Contractors

Under the authority of the Social Security Act, CMS employs a variety of contractors to process and review claims in accordance with Medicare rules and regulations. Table 2: Medicare Contractors, Responsibilities, and Contact Information lists the review contractors referenced in this implementation guide.

Table 2: Medicare Contractors, Responsibilities, and Contact Information

Type of Contractor	Responsibilities	Contact Information
Medicare Administrative Contractors (MAC)	Process claims submitted by physicians, hospitals, and other health care professionals, and submit payment to those providers in accordance with Medicare rules and regulations. This includes identifying and correcting underpayments and overpayments.	https://www.cms.gov/data-research/monitoring-programs/medicare-fee-service-compliance-programs/review-contractor-directory-interactive-map
Unified Program Integrity Contractor (UPIC)	Identify cases of suspected fraud and take appropriate corrective actions.	http://www.cms.gov/Research-Statistics-Data-and-Systems/Monitoring-Programs/Medicare-FFS-Compliance-Programs/Review-Contractor-Directory-Interactive-Map
Supplemental Medical Review Contractor (SMRC)	Conduct nationwide medical review, as directed by CMS. This includes identifying underpayments and overpayments.	http://www.cms.gov/Research-Statistics-Data-and-Systems/Monitoring-Programs/Medicare-FFS-Compliance-Programs/Medical-Review/SMRC.html
Contractor (CERT DC), CERT Review Contractor (CERT RC), and CERT Statistical Contractor (CERT SC)	Collect documentation and perform reviews on a statistically valid random sample of Medicare FFS claims to produce an annual improper payment rate.	https://www.cms.gov/Research-Statistics-Data-and-Systems/Monitoring-Programs/Medicare-FFS-Compliance-Programs/CERT/index.html?redirect=/cert
Recovery Audit Contractors (RAC)	Identify underpayments and overpayments, as part of the Recovery Audit Program.	http://www.cms.gov/Research-Statistics-Data-and-Systems/Monitoring-Programs/Medicare-FFS-Compliance-Programs/Recovery-Audit-Program/
Qualified Independent Contractor (QIC)	A party to the redetermination may request a reconsideration if dissatisfied with the redetermination decision. QIC conducts the reconsideration.	https://www.cms.gov/medicare/appeals-and-grievances/orgmedffsappeals/reconsiderationbyaqualifiedindependentcontractor.html

1.2 System Overview

The esMD system provides a mechanism for exchanging medical documentation and responses for Cross-Enterprise Document Reliable Interchange (XDR) and X12N 278

requests between the Medicare Provider community and the Medicare RC community. The purpose is to enable the electronic transmission of information between HIHs who represent Providers and the Medicare RCs in a manner that replaces paper documents where possible.

The RC Client is a utility that enables RCs to communicate with esMD by exchanging files using, Upload, Download, Notification, and Status APIs in the esMD cloud environment.

Note: The esMD system identifies submissions and requests sent from HIHs to RCs as inbound files and identifies transactions and responses for XDR and X12N 278 sent from RCs to HIHs as outbound files.

1.3 System Requirements

See Section 7 System Requirements for the system requirements for installing the Java version of the RC Client.

Section 7 System Requirements provides the requirements needed for the computer system where the RC Client will be installed, including the computer system's processor, amount of disk space and free memory needed, permissions, minimum internet connectivity Kilobits Per Second (Kbps) transfer speeds, and the Microsoft Java Framework version needed to run the RC Client properly.

Refer to the Identity Management (IDM) Instructions in the link below for details on how to obtain an IDM login:
https://www.cms.gov/data-research/cms-information-technology/cms-identity-management/guides-documentation

Refer to Section 1.4.3 RC Client Operation Overview for Enterprise File Transfer (EFT) Password requirements per IDM policy for logging in to the internal server.

https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/EnterpriseIdentityManagement/Guides-and-Documentation

1.4 RC Client Overview

esMD release AR2023.01.0, in January 2023, continued to support existing functionality and LOBs, while adding the following new enhancements as part of the esMD cloud migration:

1. Replaced the existing file transfer mechanism with esMD Upload and Download APIs to transfer the files in the esMD cloud environment.

2. Enabled RC Clients login with an Identity Management (IDM) System username and password and have the user credentials authenticated by the esMD Authentication API.

3. Implemented the Notification API to send notifications (Admin Error, Pickup, PA Reject Response etc.) to HIHs and receive the responses back in real time.

4. Implemented the Status API to run as a scheduled service to pull the delivery confirmation errors or any validation failures at specific intervals of time.

esMD release AR2023.04.0, in April 2023, continue support of existing Post-Pay functionality and added the following new capabilities:

1. The additional Review Contractor Types (QIC, CERTs & SMRCs) can now start submitting the new Post-Pay-Other Electronic Medical Documentation Requests (eMDR) to esMD application.

The esMD system enabled a new PCR program, Inpatient Rehabilitation Facility (IRF) to be received in XDR format as part of AR2023.08.0 implementation.

esMD release AR2023.10.0, in October 2023, continue support of existing functionality and LOBs added the following new capabilities:

1. The additional Review Contractor Types (MACs and RACs) can now start submitting the Prior Authorization Decision Letters and Review Result Letters to esMD application.

esMD release AR2024.04.0 in April 2024 added the following new capabilities:

1 Introduce a new National Provider Identifier (NPI) metadata element in the process metadata file. The addition of the NPI element metadata is required only for the Pre-Pay eMDR process of all claim types. The purpose of this change is to report specific errors for Pre-Pay Electronic Medical Documentation Requests (eMDRs). The participating RCs are Part A/B and DMACs.

2 esMD will start accepting the IRF PCR program in X12N 278 format. RCs will receive the RC package for IRF via the RC Client. The X12N 278 request is shared via the Shared Systems like other PA programs. No changes will be needed on the RC Client API to accept IRF X12 packages. The RCs will submit Admin and PA Reject Responses via the RC Client to the esMD system. The IRF-PCR is effective only for the MACs with a Jurisdiction/State of Palmetto Part A JJ/Alabama.

esMD release AR2024.10.0 in October 2024 added the following new capabilities:

1. Updated the existing eMDR Content Type Codes (CTCs) with the new CTCs as follows:
    a. eMDR Post-Pay and eMDR Post-Pay-Others is changed from 1.6 to 2.6.
    b. eMDR Pre-Pay is changed from 1.5 to 2.5.

2. The esMD System will update the CTC description as 'Letters' instead of 'RRL' for the Content Type Code 20.

esMD release AR2025.04.0 in April 2025 added the following new capabilities:

1. Implemented enhancements to the esMD electronic letter process (LETTERS) to incorporate the common qualifier values in the LETTERS to correctly link it to the associated case/situation it relates to.

   These changes affect only the RCs and HIHs that are currently sending and receiving LETTERS.
2. Implement the combined errors approach for PA Reject Errors 57/15 for XDR PA Reject Response scenarios.

1.4.1 RC Client Pull/Push Functionality

The RC Client provides the following functionality:

- Download:
  - Inbound documents (submitted by HIHs) from the esMD Cloud environment
  - HIH acknowledgements indicating receipt of pick-up notifications, PA Review Result Responses, Administrative Error Response, and HIH Delivery Notification
  - Data Element Validation results for the outbound process
  - ICDT Request
  - ICDT Solicited Response
  - ICDT Unsolicited Response
  - ICDT Batch Notifications (Acknowledgement/Pickup Notifications)
  - ICDT Validation Failures/Error Notifications
  - ICDT Administrative Errors
  - eMDR Service Registration Request
  - Document Code File Request
- Upload:
  - Error responses to PA Requests for XDR and X12N 278 to esMD
  - Administrative Error response for XDR and X12N 278 to esMD
  - Error messages generated due to file decompression and checksum verification
  - Acknowledgement messages for receipt of documents and authorization requests
  - Site-Specific Configuration settings:
    - Push frequency/Pull frequency.
    - Folder locations for both Inbound and Outbound files.

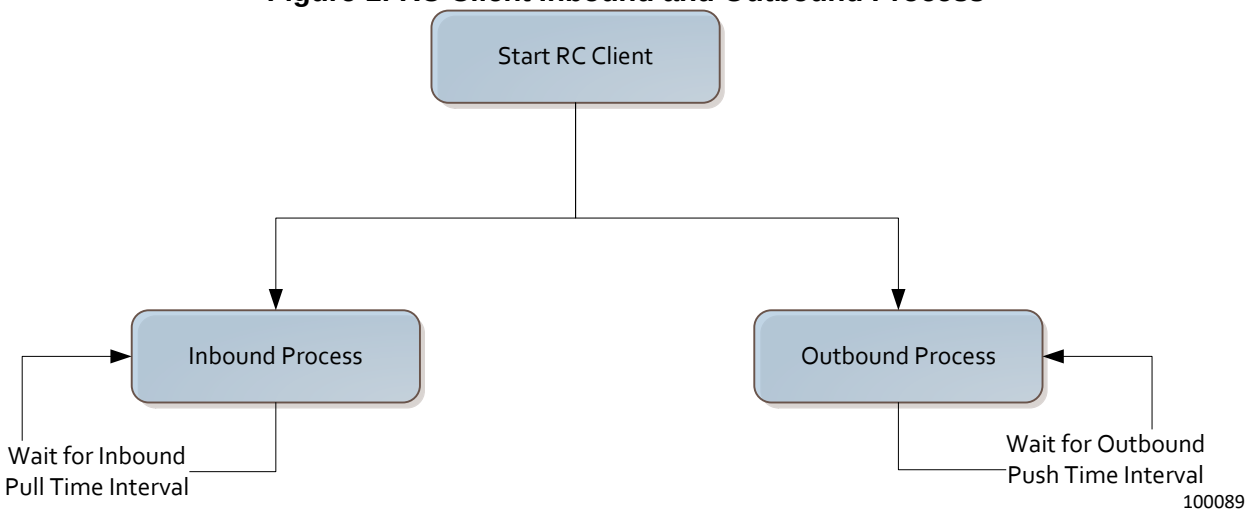
- ICDT to esMD:
  - ICDT Request
  - ICDT Solicited Response
  - ICDT Unsolicited Response.
- ICDT Pickup Notifications/Error Notifications to esMD
- ICDT Administrative Error Response to esMD
- eMDR Service Registration Request Pickup Notification
- Document Codes Request Pickup Notification.
- Letters Realtime Requests.
- eMDR Pre-Pay Requests
- eMDR Post-Pay Requests
- eMDR Post-Pay Other Requests

1.4.2 RC Client Application Overview

The esMD RC Java Client is a standalone Java Windows desktop application that runs outside the CMS network on the RC's machine, computer, or server. The purpose of the RC Java Client is to connect to the esMD cloud environment using the esMD Auth API to upload and download files. The RC Java Client uses the IDM System login credentials, and the user is authenticated using the esMD Auth API. RC Client users (at the RC site) provide their login credentials when they start the RC Client on their machines.

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 and will upload and download files automatically without continual user intervention, based on the time intervals set by the RC. The user is authenticated every time the inbound and outbound process starts and also when the files are actually uploaded and downloaded.

Figure 1: RC Client Inbound and Outbound Process



In the Inbound process, the RC Client connects to the esMD cloud environment using the Auth API and executes the Download API process to get list of available files using the token returned by the Auth API. The download process then iterates through every file from the list and invokes the esMD Download API again with the filename and token to get the pre-signed URL. The file object is then downloaded from esMD cloud using the pre-signed URL and token. The documents are pulled into the RC's inbound user directory for the authenticated user and waits for the next cycle, as determined by the Inbound Pull Time Interval setting.

In the Outbound process, the RC Client connects to the esMD cloud environment using the Auth API and executes the Upload API process to upload files to esMD cloud by using the pre-signed URL and token returned by the esMD Auth API. The RC Client waits for the next cycle as determined by the Outbound Push Time Interval setting.

The Upload Realtime API process uploads the Letters JSON messages to esMD using the token generated by Auth API. esMD processes the request and sends the response (Validation Error or HIH Delivery notification) in real time. esMD also has a deferred approach to process Letters requests if the size of the JSON message is greater than 10 MB.

The inbound pull time interval is independent of the outbound push time interval.

Note: Running multiple instances of the Java RC Client for the same jurisdiction could result in errors while pulling the files.

The RC does not need to log in to the esMD cloud environment in order to create Error Responses and Administrative Error Responses. The login is necessary only to download or upload files from or to esMD cloud. The Notification API is used to process the PA Reject and Administrative Errors which internally use the encrypted user credentials stored in memory and invokes the Auth API to get the user authenticated.

1.4.3 RC Client Operation Overview

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 Application is interrupted by two events:

1. When IDM passwords expire (**Note: IDM passwords expire every 60 days, if not changed**).
2. When a Virus Scan/Infected File error notification is received from the esMD System.

In the first scenario, when the IDM 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 esMD cloud environment. The IDM System notifies the user 15 days prior to the password expiring. For more information on the IDM User Credentials and how to reset the password, please refer to the IDM Instructions document in the esMD Downloads section, using the link below:

http://www.cms.gov/Research-Statistics-Data-and-Systems/Computer-Data-and-Systems/ESMD/Information_for_Review-Contractors.html

The password setup in the portal must meet the following IDM for users to be able to log into Internet Server:

PASSWORD POLICY

1. Passwords must be at least 15 characters in length.
2. Passwords must include an uppercase letter.
3. Passwords must include a lowercase letter.
4. Passwords must include a number (0 - 9).
5. Passwords must not contain a space.
6. Passwords must not be one of the user’s last 6 passwords.
7. Passwords must not contain parts of the user’s First Name, Last Name, or User ID.
8. 24 hours must have elapsed since the last password change.

Note: After the password reset, update the password to the new password in the configuration or script file if it is being stored and used by RC Client.

In the second scenario, when a Virus Scan/Infected File error notification with the delivery type ‘X’ is received from esMD by Status API, the RC Client application is

terminated. The RC is advised to contact the esMD Service Desk (refer to Section 29. Contacts for more details) on any Virus Scan Notifications.

1.5 ICDT Overview

ICDT functionality enables RCs to route ICDT Requests and ICDT Solicited/Unsolicited Responses to other RCs.

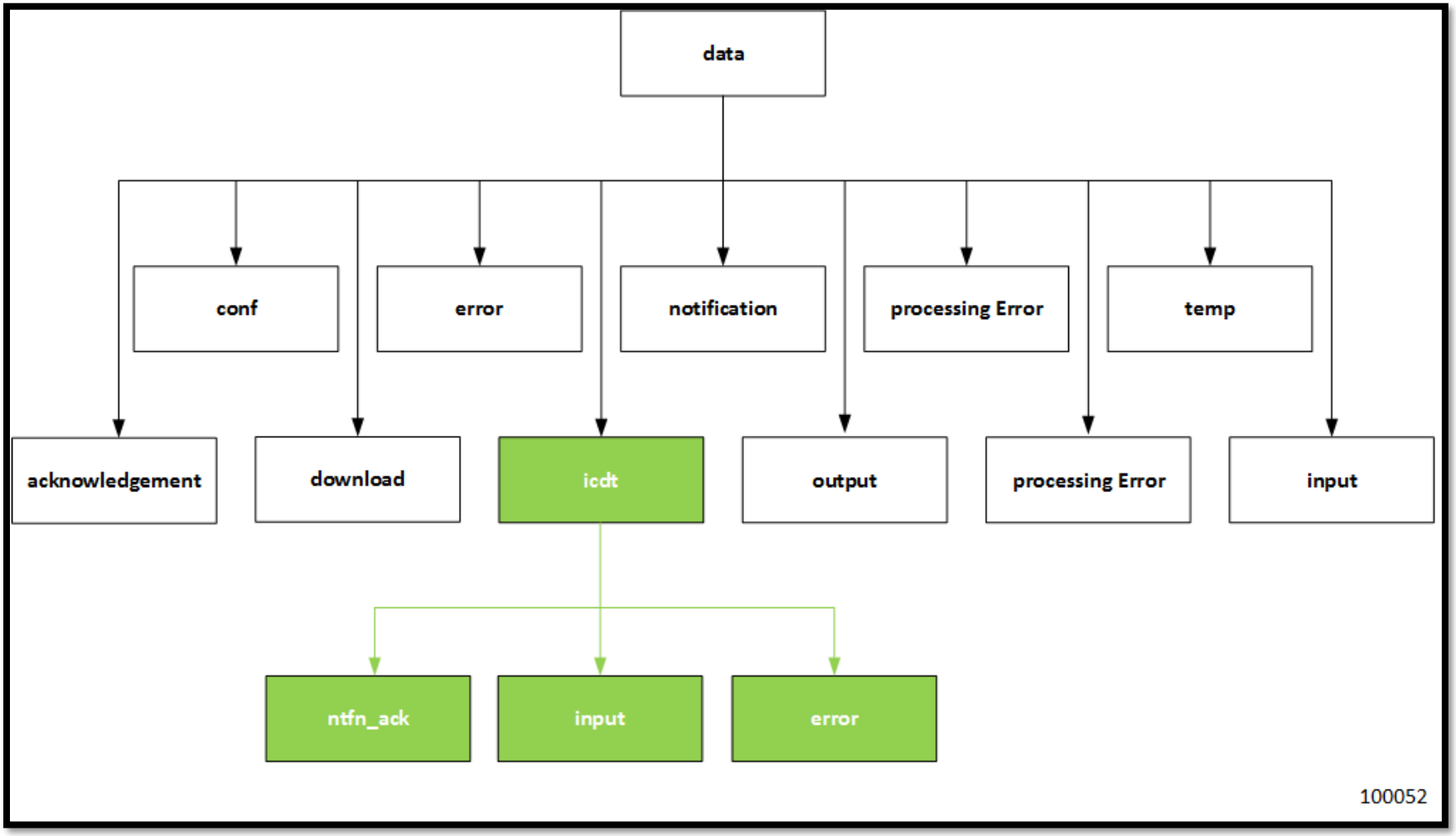
esMD supports the following types of ICDT Requests and ICDT Solicited/Unsolicited Responses as part of the initial pilot program:

1. ICDT Request/Solicited Response: RC-A sends an ICDT Request to RC-B requesting certain documentation for a claim or a case, and RC-B responds (ICDT Solicited Response) to RC-A with the requested attachments.
2. ICDT Unsolicited Response: RC-A sends the ICDT Unsolicited Response documentation bundle to RC-B (e.g. misdirected documentation).
3. RCs download the ICDT Request/Solicited Response and sends the pickup notification using Notification API to esMD in real time.
4. The ICDT Status API in the RC Client application runs at specified time intervals to retrieve the ICDT related statuses (pickup ack, admin error, validation error etc..) from esMD and deliver them to the RC.

1.5.1 RC Client ICDT Folder Structure

A separate folder structure is used for placing the ICDT Request/Solicited Response, Unsolicited Response, Notifications, errors, and Acknowledgments files. The older ‘icdt’ is under the ‘data’ folder as shown Figure 2: RC Client ICDT Folder Structure. The ‘icdt’ folder contains three folders. The ICDT Request and ICDT Response files are moved to the ‘input’ folder. All the notifications and acknowledgments are placed in the ‘ntfn_ack’ folder. Any validation errors and admin errors received from the esMD system are moved to the ‘error’ folder.

Figure 2: RC Client ICDT Folder Structure



2. Overview of How This Document is Structured

This document is structured into the following two primary sections.

1. First primary section of this document provides the following:

   - How to start and log into the RC Client.
   - How to enter a Review Response decision.
   - How to enter an error code for a PA request.
   - How to submit Inbound Submissions errors.
   - Advanced debugging, which shows how to test to determine if the RC Client application can connect to the esMD Cloud environment and to determine if there are any inbound files ready for downloading.

   ❖ The audience for this first section is the RC business users.

2. How to install and configure a Java version of RC Client.

   ❖ The audience for this second section is the person(s) installing the RC Client application.

   This section provides the technical specifications for installing and configuring RC Client on a computer system or network and includes the following:

   - Overview of the installation process.
   - Systems Requirements for a Java installation.
   - Installing an Out-of-Box Java version of the RC Client application.
   - File transfers from esMD Cloud environment.
   - XML/JSON Messages, including Outbound, Inbound, and Error messages.
   - Inbound Processes and Files.
   - Outbound Processes and Files.
   - Configuring the RC Client application.
   - RC Client Components.
   - RC Client Workflow.

- RC Client application Utilities, Components, Schedulers, and Encryption.
- Changes to the API.
- Using API.
- Configuring the RC Client application for notifications.
- Processing and pulling in documents.
- Security.

3. How to Start the RC Client and Log In

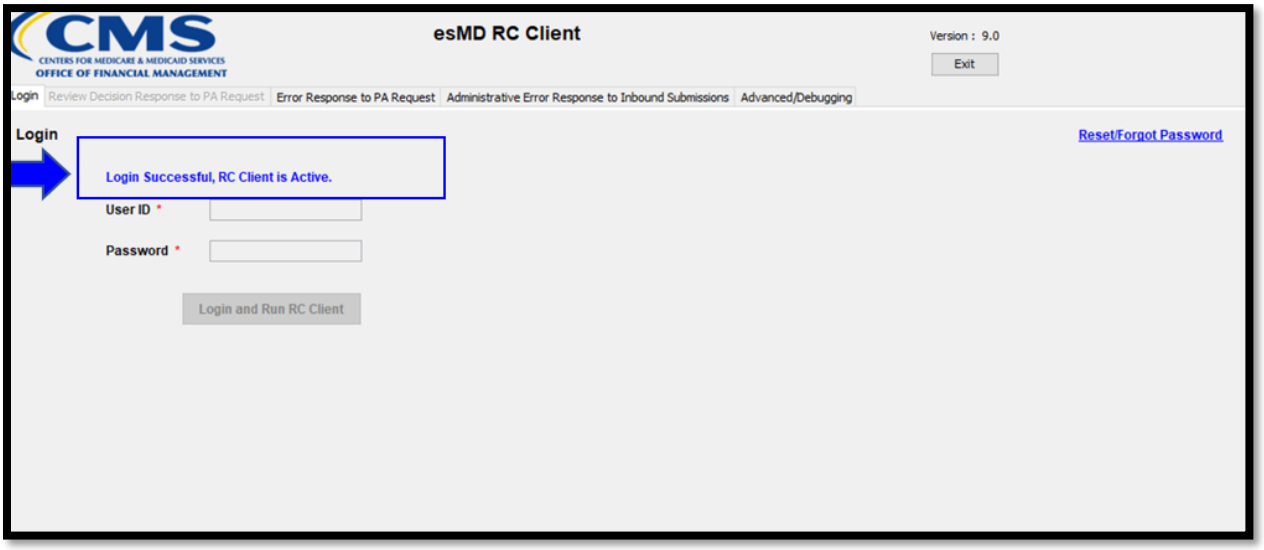
The following are the step-by-step instructions for starting the RC Client and logging in.

Step	Action
Step 1. Starting the RC Client and Logging In	Start the RC Client by selecting the rcclient.bat in the RC Installation folder or directory.
Step 2. Starting the RC Client and Logging In	The Login screen is displayed. Enter your IDM User ID and password, then select Login and Run RC Client. ❖ Note: The IDM login credentials are confidential and should not be shared with others. (For more information on IDM login credentials, see IDM's User Guide here: (Guides & Documentation | CMS) . The user should also update the valid ClientID and ClientSecret values in the esmd-rc-client-config.xml file for the user to be able to login.

Figure 3: RC Client Login Screen

Step	Action
Step 3. Starting the RC Client and Logging In	After a successful log in, the Login Successful, RC Client is Active. message is displayed.

Figure 4: RC Client - Login Successful Message

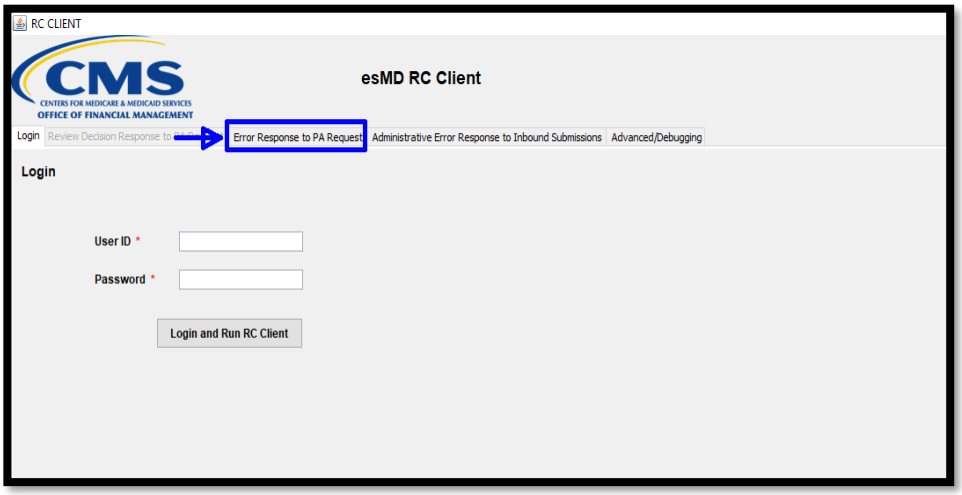


4. How to Enter an Error Code on the Error Response to PA Request Tab

This section provides step-by-step instructions on how to enter an error code on the Error Response to PA Request tab.

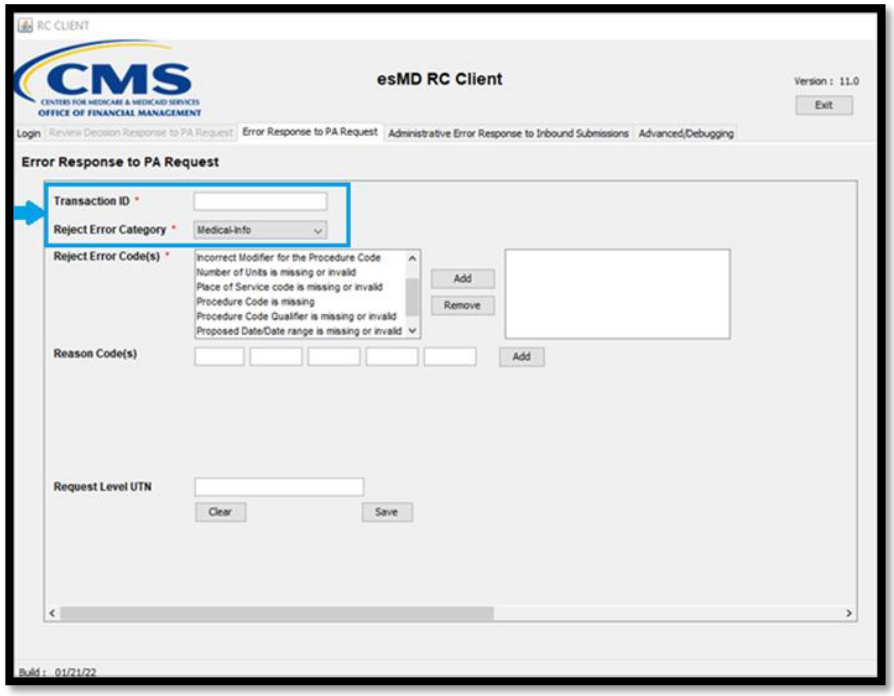
Step	Action
Step 1. Entering an **Error Code**	Select the Error Response to PA Request tab. ❖ After a successful log in, another log in is not required to navigate to and use the Error Response to PA Request tab.

Figure 5: RC Client - Error Response to PA Request Tab



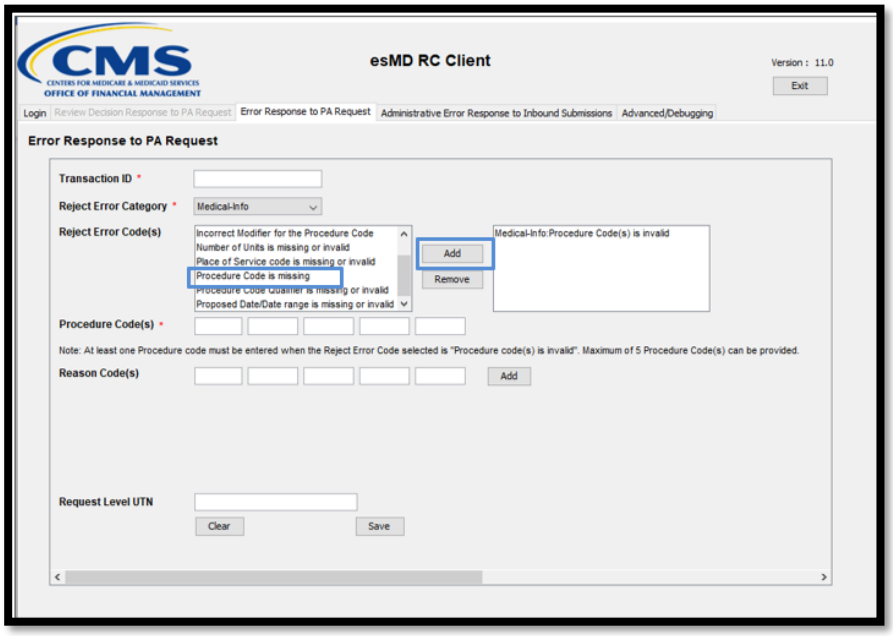
Step	Action
Step 2. Entering an **Error Code**	The fields for the Error Response to PA Request tab are displayed. ❖ Before You Begin: If you need a brief description of any of the fields on the tabs, see Appendix A: Description of Fields on RC Client Tabs. Enter the Transaction ID and select a Reject Error Category.

Figure 6: RC Client - Error Response to PA Request Reject Error Category



Step	Action
Step 3. Entering an **Error Code**	Select a Reject Error Code and the Add button to add the Reject Error Code. For information on how to access an up-to-date list of Reject Error Codes, see Appendix B: Reject Error Codes.

Figure 7: RC Client - Error Response to PA Request Reject Error Codes



Step 4. Entering an **Error Code**	Enter the Reason Code(s). Select the Add button at the end of the row of the Reason Code field to add additional rows of Reason Codes, as needed. (Note: This step is optional.) Enter the Request Level UTN and select Save to submit the Error Code for submission.

Technical Note: After selecting Save, the Notification API creates a PA Reject Response JSON message and sends it to the esMD system after authenticating the user with the Auth API. The PA Reject Response is delivered to the HIH and the delivery confirmation in JSON format is returned to the RC Client. The notification API process converts the JSON message to xml file and saves in the notification folder.

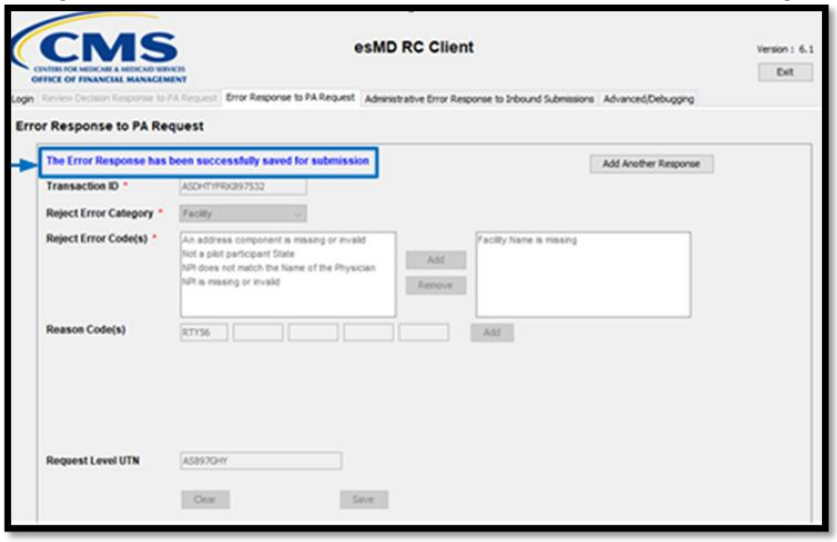
Step	Action
Step 5. Entering an **Error Code**	After selecting Save, “The Error Response has been successfully saved for submission” message is displayed.

Technical Note: After selecting Save, the RC Client validates the data entered and displays errors messages, as applicable.

If the data validation is successful, the Error Code is created, and the “The Error Response has been successfully saved for submission” message is displayed.

Note: After successfully saving a decision for submission, all information in the fields is cleared, and other responses can be entered.

Figure 8: RC Client - Error Response Submission Success Message

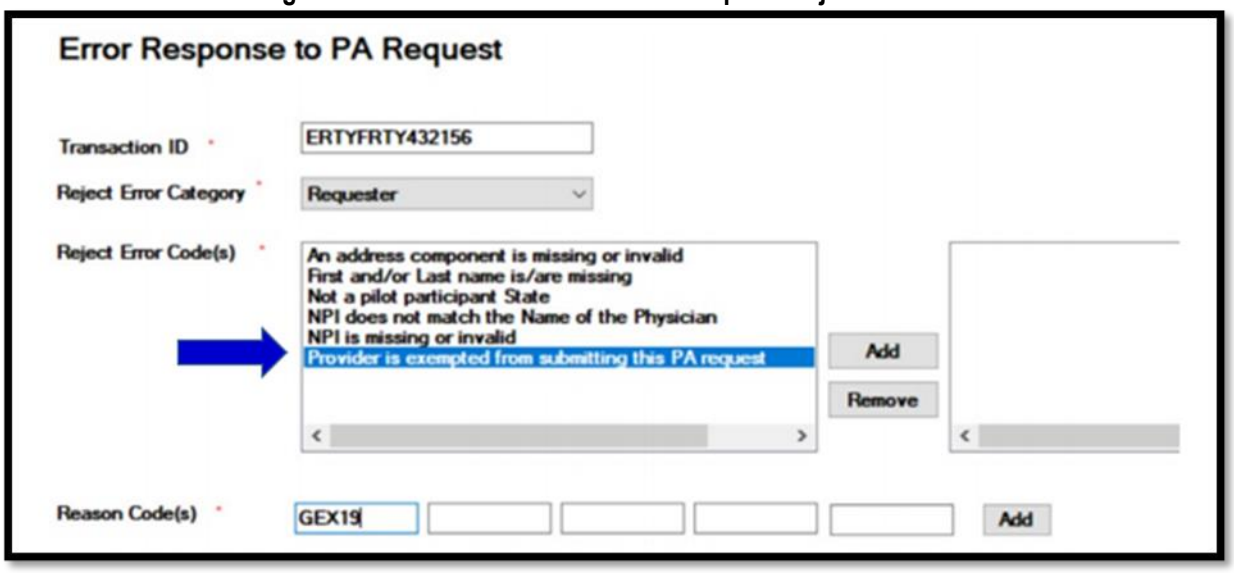


The Error Response to PA Request screen on the RC Client user interface (UI) was updated to add a new Reject Error Code, which allows RCs to reject PA/PCR requests for Provider Exemption. The new Reject Error Code was added under the Requester Category of the Reject Error Category. In addition, esMD started accepting the GEX19 Reason Code from the RCs as part of the Reject responses.

The reject error category ‘Medical Info’, the following Reject Error Code(s) are sent to the HIH through the RC Client in the X12N 278 response:

- ‘15 – Number of Units is missing or invalid’
- ‘57 – Proposed Date/Date Range’.

Figure 9: RC Client - New Provider Exemption Reject Error Code

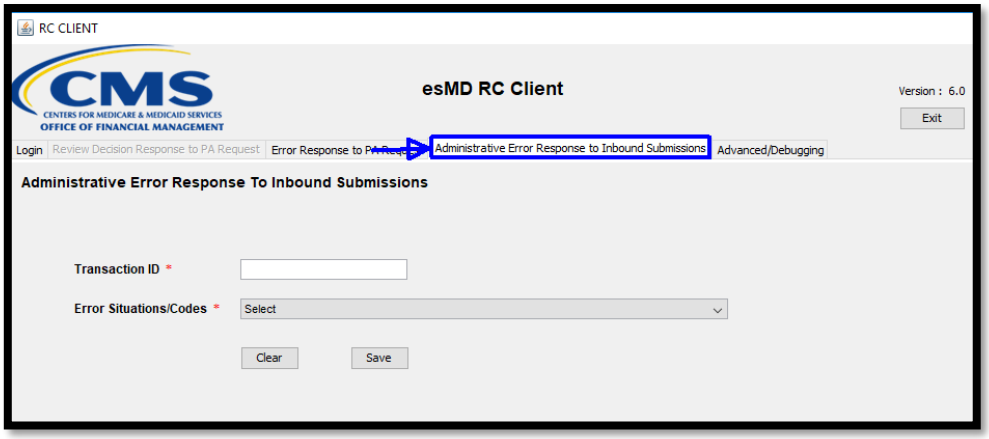


5. How to Submit an Inbound Submission Error on the Administrative Error Response to Inbound Submissions Tab

This section provides step-by-step instructions on how to enter an inbound submission error on the Administrative Error Response to Inbound Submissions tab.

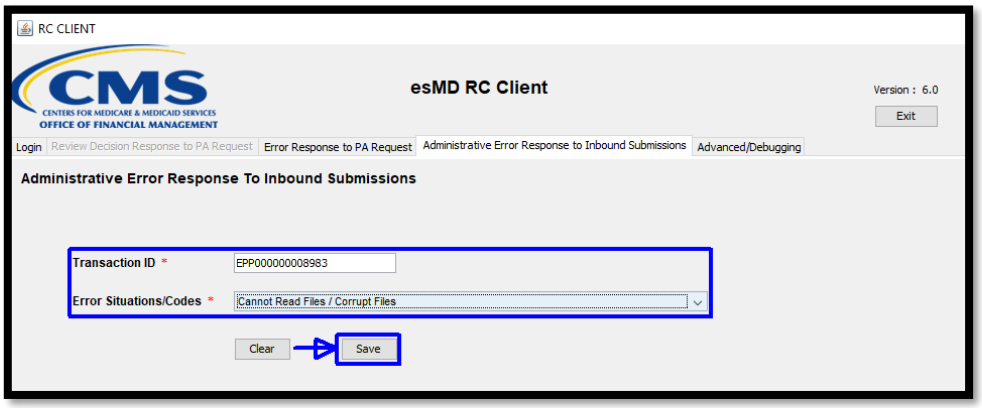
Step	Action
Step 1. Entering an **Inbound Submissions Error**	Select the Administrative Error Response to Inbound Submissions tab. ❖ After a successful log in, another log in is not required to navigate to and use the Administrative Error Response to Inbound Submissions tab.

Figure 10: RC Client - Admin Error Response Screen



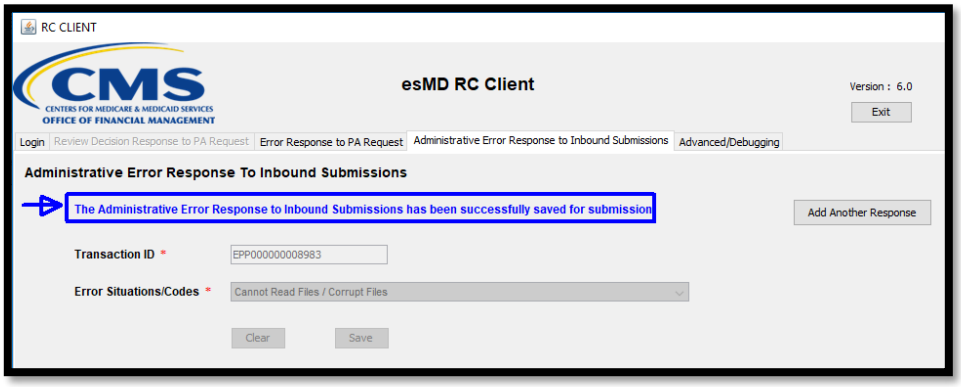
Step	Action
Step 2. Entering an **Inbound Submissions Error**	The fields for the Administrative Error Response to Inbound Submissions tab are displayed. ❖ Before You Begin: If you need a brief description of any of the fields on the tabs, see Appendix A: Description of Fields on RC Client Tabs. Enter the Transaction ID, select an Error Situation or Error Code from the Error Situations/Codes drop down menu, and then select Save to submit the Inbound Submissions error for submission. Technical Note: After selecting Save, the Notification API invokes the Auth API to authenticate the user using the encrypted user credentials stored in memory. After authenticating the user, an Admin Error JSON message is created and sent to esMD. The Admin Error is delivered to the HIH and the delivery confirmation in JSON format is returned to the RC Client. The notification API process converts the JSON message to an XML file and saves in the notification folder.

Figure 11: RC Client - Save Admin Error Response



Step	Action
Step 3. Entering an **Inbound Submissions Error**	After selecting Save, the "The Administrative Error Response to Inbound Submissions has been successfully saved for submission" message is displayed. Technical Note: After selecting Save, the RC Client validates the data entered and displays errors messages, as applicable. If the data validation is successful, the Inbound Submissions Error is created, and the "The Administrative Error Response to Inbound Submission has been successfully saved for submission" message is displayed. Note: After successfully saving a decision for submission, ***Add Another Response button can be used to submit another response.***

Figure 12: RC Client - Admin Error Response Success Message

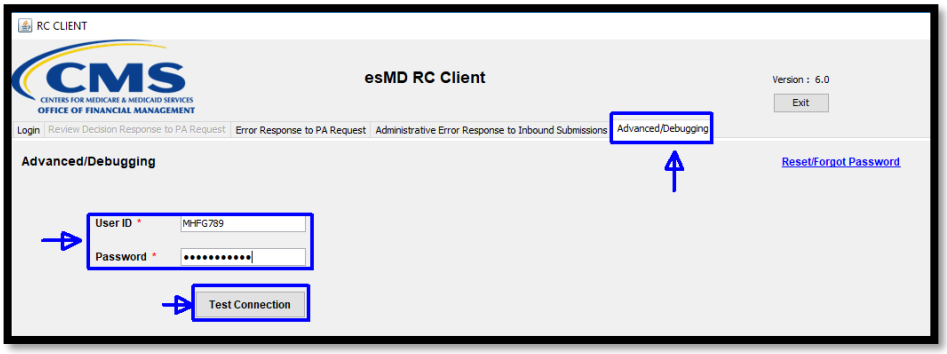


6. How to Verify Connection to esMD Cloud Using Advanced/Debugging Tab

This section provides step-by-step instructions on how to verify connection to the esMD Cloud, using the Advanced/Debugging tab.

Step	Action
Step 1. Checking Connection to esMD Cloud	Select the Advanced/Debugging tab. The Advanced/Debugging tab fields are displayed. On the Advanced/Debugging tab, enter your IDM User ID and password. (This is required on the Advanced/Debugging tab.) Select Test Connection.

Figure 13: RC Client - Advanced Debugging Tab



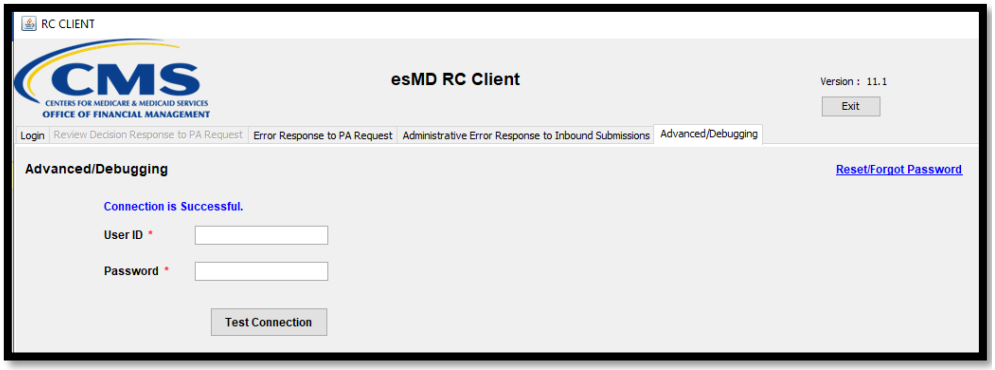
Step 2.

Checking Connection to esMD Cloud

After selecting Test Connection, the “Connection is Successful.” message is displayed.

Note: After successfully testing your connection, you may select another tab.

Figure 14: RC Client - Debugging Successful Message



7. System Requirements

The following are the system requirements for installing a Java version of the RC Client.

7.1 Processor

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

7.2 Disk Space

The disk requirement for the RC Java Client is 10 Megabytes (MB). The documents that the RC Client pulls from the esMD Cloud may require additional disk space.

7.3 Memory

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

7.4 Permissions

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

7.5 Network

The RC Client requires internet connectivity that supports more than 32-Kbps transfer speeds.

7.6 Java Framework

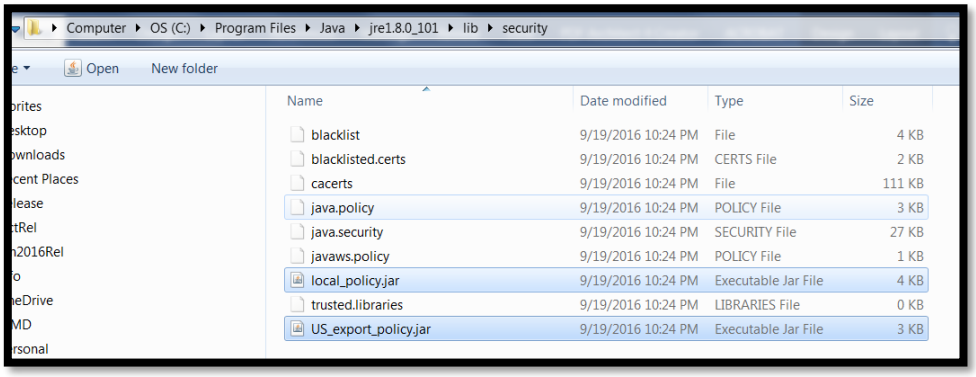
The RC Client requires Java Runtime Environment (JRE) 1.8 or greater. Section 7.6.1 describes how to download and configure the Java RC Client for upgrading the Java version to 1.8.

7.6.1 How to Configure the RC Client for Java 1.8 Version from an Earlier Version

Step	Action
Step 1.	Download Java 1.8 (Java SE Development Kit 8u101) from the Oracle website and install it on the machine where RC Client has been set up.

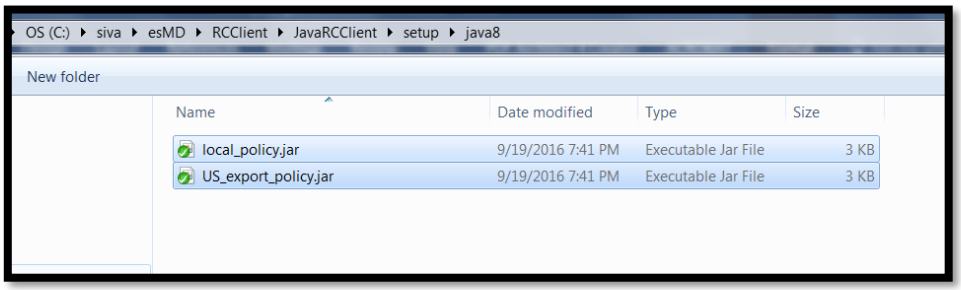
Step	Action
Step 2.	Backup the following Java Cryptography Encryption (JCE) Policy Files from the "...\Java\jre1.8.0_101\lib\security" path (refer to Figure 15: Directory Structure for the Policy Files): 1. local_policy.jar 2. US_export_policy.jar Delete these two files after backing them up.

Figure 15: Directory Structure for the Policy Files



Step 3.	Copy the JCE Policy Files (local_policy.jar, and US_export_policy.jar) provided in the RC Client installation package from "...\esMDRCClientAPISampleClient\setup\java8" as shown in Figure 16: Setup Directory for Java, and place it in the same path mentioned in step 2 ("...\Java\jre1.8.0_101\lib\security").

Figure 16: Setup Directory for Java



Step 4.	Run the RC Client and login.

7.7 Libraries

Table 3: Libraries lists all third-party libraries used by the RC Client along with their corresponding versions and a brief description of how the RC Client uses them.

Table 3: Libraries

Library	Version	Description
commons-codec	1.7	Used for Encoding and Decoding
commons-compress	1.20	Used for Extraction and compression of the packages.
commons-io	2.4	Used for reading and writing files to the Filesystem.
commons-lang	2.6	Used by the Java Secure Channel for Helper Utilities
commons-logging	1.1.1	Logging Framework used by the Jsch.
Jcalendar	1.4	Used for the popup Calendar in the GUI
log4i-core	2.17.2	Logging Framework
Log4j-api	2.17.2	Logging Framework
aws-core	2.17.209	AWS core Libraries
Encryption	1.0	Used to encrypt and decrypt passwords
Httpclient	4.5.13	Used by the API's for authentication,upload and download files
HttpCore	4.4.15	Used by the API's for authentication,upload and download files
json	20220320	Used for processing JSON messages.
JSON-simple	1.1.1	Used for processing JSON messages

8. How to Install and Configure a Java Version of the RC Client

Review the System Requirements in Section 7. System Requirements to make sure the machine that will host the RC Client meets the necessary requirements.

You can install the RC Client in two ways:

1. Out of the box.
2. Custom RC Client (Java).

8.1 Out-of-the-Box

The RC Java Client API comes packaged with a sample client. To run this sample client out-of-the-box, the RCs need to follow the procedures in the following sections.

8.1.1 Download RC Client

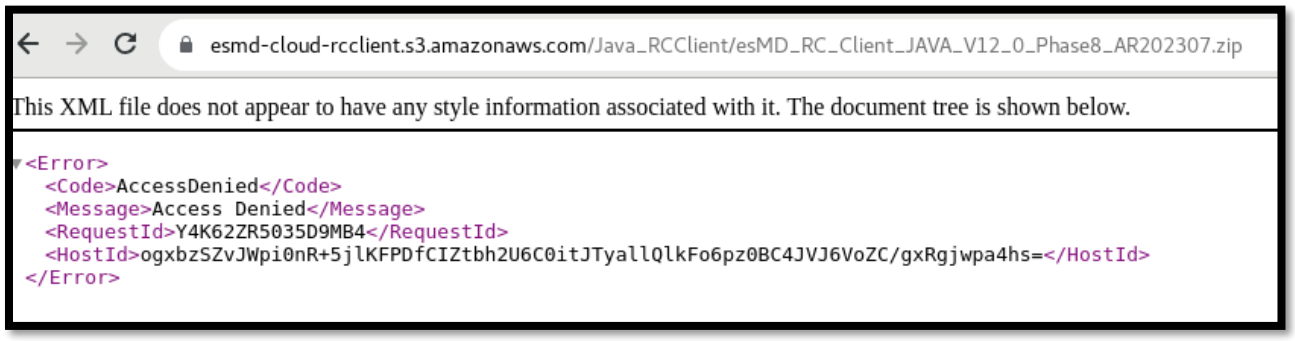
The RC Client software is securely accessible only to systems with whitelisted IP addresses. This stringent measure guarantees that the RC client can be downloaded and installed exclusively on machines running on authorized IP addresses. By restricting access to trusted networks, we prioritize the safety and integrity of the RC Client, ensuring that it is available precisely where it is intended to be used.

Download the RC Client by clicking on the link below.

https://esmdcloud2-uat.cms.hhs.gov:8089/downloadrc/esMD_RC_Client_JAVA_V14_Phase1_AR202404.zip

If your IP is not whitelisted, you will encounter the "Access Denied" message, which indicates that access has been restricted. In such a situation, we kindly request you reach out to the esMD Ops team for prompt assistance and resolution.

Figure 17: Access Denied Error Message



8.1.2 KeyStore Set Up

Important: The RC Client uses asymmetric encryption to store the IDM user credentials securely. For this encryption to work, you will 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. Please refer to Section 24.1 Security for more details on the Security framework used by the RC Client.

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.

```
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
```

Note: Replace <parameter> with the value of the parameter from the list in Table 4: Keystore Creation Parameters.

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

Important: After the Public and Private keys have expired, you must re-create both keys to continue to use the RC Client.

Table 4: Keystore Creation Parameters

Where	Means
<keystore>	The keystore is the home location. If you do not specify the <keystore> option, the default keystore file named *keystore.jks* in the user's home directory will be created, if it does not already exist. For example, the config/keystore.jks will be created.
<alias>	The certificate chain and the private key are stored in a new keystore entry, identified by *alias.*
<storepassword>	The store password is used to protect the integrity of the keystore. It must be at least six characters long.
<keypassword>	The key password is used to protect the private key of the generated key pair. If a password is not provided by the user, the user is prompted to provide it. If you press Enter at the prompt for the key password, the key password is set to the same password that was used for the keystore. The <keypassword> must be at least six characters long.
<commonName>	The common name is the name for any entity, such as the name of a person (for example, Susan Jones) or the name of your company.
<organizationalUnit>	The organizational unit can be used for a small organization, department, or division of an organization (for example, Purchasing).
<organizationName>	The organization name is for a large organization or company (for example, ABC Systems, Inc.).
<localityName>	The locality name can be for a city (for example, Palo Alto).

Where	Means
<stateName>	The state name can be for a U.S. state or province of another country (for example, California or Ontario in Canada).
<country>	The country is a two-letter code (for example, US).

8.1.3 Integrity Verification

The following command 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>
```

Note: Replace <parameter> with the value for the parameter listed in Table 4: Keystore Creation Parameters.

8.1.4 Java Cryptography Extension (JCE) Policy Update

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

8.1.4.1 Understanding the JCE Security Policy Files

Due to import control restrictions, the version of the JCE security policy files bundled in the Java Development Kit™ (JDK) environment allows "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 Oracle’s JDK.

8.1.4.2 Understanding the Export/Import Issues

JCE for JDK has been through the U.S. export review process. The JCE framework, along with the Oracle 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.

8.1.4.3 JCE Policy Files

The setup directory in the RC Client installation contains the policy files listed in Table 5: JCE Policy Files.

Table 5: JCE Policy Files

Policy File	Description
local_policy.jar	Unlimited strength local policy file
US_export_policy.jar	Unlimited strength U.S. export policy file

8.1.4.4 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 6: Java Development Kit and Table 7: Java Runtime Environment show examples of the installation for Java version 1.8.

Table 6: Java Development Kit

Environment	Example JDK Installation Directory	JAVA_HOME
Windows	C:\jdk1.8.0	C:\jdk1.8.0\jre
Unix	/home/user1/jdk1.8.0	/home/user1/jdk1.8.0/jre

Table 7: Java Runtime Environment

Environment	Example JRE Installation Directory	JAVA_HOME
Windows	C:\jre1.8.0	C:\jre1.8.0
Unix	/home/user1/jre1.8.0	/home/user1/jre1.8.0

Notes:

1. UNIX (Solaris/Linux) and Windows use different path name separators; use the appropriate one ("\" or "/") for your environment.

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 Java Archive (JAR) files for all JREs that you plan to use.

8.1.4.5 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.

2. Copy the policy files with the unlimited strength versions from the “setup” directory per the version of Java to be used under the installation directory to the security directory shown in Table 8: Security Directory.

Table 8: Security Directory

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

8.1.5 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).

   Important: The 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. ESMDSFTPServer in the config file is replaced with ESMDAuthAPI configuration which is used by esMD Auth API to connect to esMD Cloud.

   Note: Contact Helpdesk to get the ClientID, ClientSecret, hostname used by the API’s.

3. Use the comments for each configuration parameter shown in Figure 18: Sample RC Client Configuration File as a guide in entering your data.

Figure 18: Sample RC Client Configuration File

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:ESMDConfig 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 Authentication Server Configuration-->
    <ESMDAuthAPI>
        <!-- Update: Use V for VAL, T for UAT,P for PROD-->
        <environment>T</environment>
        <!--Client ID to connect to Auth server-->
        <!-- Update: The Client ID-->
        <ClientID> 3fis5elmln49c3fklkfs5v20ml </ClientID>
       <! --Client Secret to connect to Auth Server -->
        <!-- Update: The Client Secret-->
        < clientSecret>1ad4odlhll6jur6qfbaoa71o52rt8< clientSecret>
    </ESMDAuthAPI>
    <!--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>30</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 acknowledgments directory for routing the inbound
notifications from esMD/HIH-->

<acknowledgmentsDirectory>/RCClient/data/acknowledgment</acknowledgmentsDirectory>
        <!-- Update: The notifications directory for routing the inbound
notifications from esMD/HIH-->

<notificationsDirectory>/RCClient/data/notification</notificationsDirectory>
        <!-- Update: The Processing Error directory for routing only the
unprocessed notifications from esMD/HIH-->

<processingErrorDirectory>/RCClient/data/processingError</processingErrorDirectory>
        <!-- Update: The Remote Inbound Directory path on the TIBCO MFT Server.
IMPORTANT: Replace ES#### with your own mail box number-->
        <remoteInboundDir>/ES####</remoteInboundDir>
        <!--Update: The mail box number for the inbound files used to pick the
inbound files to pull-->
        <inboundRoutingId>ES####</inboundRoutingId>
        <ICDTDirectoryStructure>
            <inputDirectory>/data/icdt/input</inputDirectory>
            <errorDirectory>/data/icdt/error</errorDirectory>
            <notificationsDirectory>/data/icdt/ntfn_ack</notificationsDirectory>
        </ICDTDirectoryStructure>
```

```
<eMDRRegistrationDirectory>/data/eMDRRegistration</eMDRRegistrationDirectory>
    </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>
        <!--The Remote Outbound mail box number to push files onto esMD cloud.
LEAVE IT AS IT-->
        <outboundRoutingId>ESMD2</outboundRoutingId>
        <!--The Outbound File name prefix-->
        <outboundFilePrefix>ON</outboundFilePrefix>
    </OutboundConfig>
</ns2:ESMDConfig>
```

4. The api.properties file contains all the information used by APIs which is required to connect to esMD cloud environment.

Figure 19: Sample RC Client API Properties File

```
DOWNLOAD_SCOPE=rc/download
UPLOAD_SCOPE=rc/upload
NOTIFICATION_SCOPE=rc/notification
STATUS_SCOPE=rc/status


DEV_API_BASE_URL=https://dev.cpiapigateway.cms.gov/api/esmd/v1
VAL_API_BASE_URL=https://val.cpiapigateway.cms.gov/api/esmd/v1
UAT_API_BASE_URL=https://val.cpiapigateway.cms.gov/api/esmd/ext/v1
PROD_API_BASE_URL=https://cpiapigateway.cms.gov/api/esmd/ext/v1

AUTH_URL=/auth/generate
UPLOAD_URL=/objects
DOWNLOAD_URL=/objects
PICKUP_NOTIFICATION_URL=/objects/notification/pickup
PAREJECT_NOTIFICATION_URL=/objects/notification/pareject
LETTERS_REALTIME_URL=/objects/realtime

ADMINERROR_NOTIFICATION_URL=/objects/notification/admin

NOTIFICATION_STATUS_URL=/objects/status/rc
ICDT_NOTIFICATION_STATUS_URL=/objects/status/icdt
EMDR_NOTIFICATION_STATUS_URL=/objects/status/emdr
ICDT_ADMINERROR_NOTIFICATION_URL=/objects/notification/icdt/admin
ICDT_PICKUP_NOTIFICATION_URL=/objects/notification/icdt/pickup
```

8.1.5.1 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, as shown in Figure 20: Keystore Password Encryption and Figure 21: Private Key Password Encryption, and click OK in each Input window.

Figure 20: Keystore Password Encryption

Figure 21: Private Key Password Encryption

3. Update the XML configuration file parameter "certAlias" with the alias of the certificate you created in Section 8.1.2 KeyStore Set Up.

   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.

8.1.6 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.

1. To run the sample RC Client, run the "rcclient.bat" utility provided in the distribution package.
2. Start the RC Client by providing login credentials for the Login tab and select the “Login and Run RC Client” button.

8.2 Custom RC Client

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

1. Log in to the esMD Cloud environment using esMD Auth API (See Section 14 Auth API).
2. Get Notifications from the esMD Notifications API. (Refer to Section 17 Notification API).
3. Decrypt/encrypt and store the login credentials using a secure RSA algorithm. (Refer to Section 24.2.8 Utilities - Encryption).
4. Pull medical documentation from the esMD Cloud. (Refer to Section 16 Download API).
5. Extract the downloaded packages. (Refer to Section 16 Download API)
6. Check the payloads using checksums in the metadata. (Refer to Section 24.2.2 Download).
7. Push the outbound files from the “output” directory. (Refer to Section 24.2.3 Upload).

Note: The procedures for customizing the RC Client API are beyond the scope of this document. (The source code that will be packaged along with the RC Client contains the documentation needed for integrating the API.

9. Inbound/Outbound Filename Format

Table 9: Inbound and Outbound Files Format lists the zip/XML files that will be transferred between esMD and the RCs.

Notes:

1. ES0001 is a sample mailbox number that is used to identify the RC, and "YSQ0002051701EC" is a sample fifteen-character esMD transaction ID.
2. The esMD Transaction ID is included in the zip file name and also in the RC metadata XML file.

Table 9: Inbound and Outbound Files Format

Type	Example File Name	Description
Inbound	<<ReceiverRoutingId>> .T.L<<CTC>>. E<<TransactionID>>.<<SenderRoutingID>>.DMMddyy. THHmmssS.zip	Submissions received from esMD to the RC: • T – Environment ID. T is for User Acceptance Testing (UAT) and P is for Production (PROD). • <<ReceiverRoutingId>> – RC Routing ID • L – The Line of Business • <<CTC>> – Content Type Code (CTC) of the program • E – Delivery type of the inbound request • <<TransactionID>> – 15 character esMD Transaction ID • <<SenderRoutingID>> – Sender Routing ID which is ESMD2 • DMMddyy – Date format in MMDDYY • THHmmssS – Time format in THHmmssS
Inbound	<<DeliveryType>>_L<<CTC>>_<<esMDTransactionID >>Receipt_Acknowledgement.xml	EMDR Acknowledgments received from esMD to the RC: • L – The Line of Business • <<CTC>> – CTC of the program • A – Delivery type of the Acknowledgments • <<esMDTransactionID>> – 15 character esMD Transaction ID

Type	Example File Name	Description
Inbound	<<DeliveryType>>_L<<CTC>>_<<esMDTransactionID>>Delivery_Acknowledgement.xml	HIH delivery notification from esMD to RC: • L – The Line of Business • <<CTC>> – CTC of the program • N – Delivery type of the HIH Notifications • <<esMDTransactionID>> – 15 character esMD Transaction ID
Inbound	<<DeliveryType>>_L<<CTC>>_<<esMDTransactionID>>Validation_Error.xml	Any validation failures from esMD: • L – The Line of Business • <<CTC>> – CTC of the program • F – Delivery type of the esMD validation failure • <<esMDTransactionID>> – 15-character esMD Transaction ID
Inbound/ Inbound	<<ReceiverRoutingId>>.T.L<<CTC>>.Q<<TransactionID>>.<<SenderRoutingID>>.DMMddyy.THHmmssS.zip	ICDT Solicited Request from RC to esMD to RC: • T – Environment ID. T is for UAT and P is for PROD • <<ReceiverRoutingId>> – RC Routing ID • L – The Line of Business • <<CTC>> – CTC of the program • Q – Delivery type of the ICDT Solicited request • <<TransactionID>> – 15-character esMD Transaction ID • <<SenderRoutingID>> – Sender Routing ID which is ESMD2 • DMMddyy – Date format in MMDDYY • THHmmssS – Time format in THHmmssS.

Type	Example File Name	Description
Inbound/ Outbound	<<ReceiverRoutingId>>.T.L<<CTC>>.R<<TransactionID>>.<<SenderRoutingID>>.DMMddyy.THHmmssS.zip	ICDT Solicited and Unsolicited Response from RC to esMD to RC. • T – Environment ID. T is for UAT and P is for PROD • <<ReceiverRoutingId>> – RC Routing ID • L – The Line of Business • <<CTC>> – CTC of the program • R – Delivery type of the ICDT Solicited and Unsolicited Response • <<TransactionID>> – 15-character esMD Transaction ID • <<SenderRoutingID>> – Sender Routing ID which is ESMD2 • DMMddyy – Date format in MMDDYY • THHmmssS – Time format in THHmmssS.
Inbound/ Outbound	<<PackageUniqueID>>_icdtadmin.json	ICDT Administrative error Response from RC to esMD to RC: • L – The Line of Business • <<PackageUniqueID>> – 15-character RequestID or ResponseID •
Inbound/ Outbound	<<ReceiverRoutingId>>.T..Q<<TransactionID>>.<<SenderRoutingID>>.DMMddyy.THHmmssS.zip	ICDT Solicited Request from RC to esMD to RC and ICDT acknowledgments from esMD to RC: • T – Environment ID. T is for UAT and P is for PROD • <<ReceiverRoutingId>> – RC Routing ID • ICDT – Type of response • Q – Delivery type of the ICDT Solicited Request • <<TransactionID>> – 15-character esMD Transaction ID • <<SenderRoutingID>> – Sender Routing ID which is ESMD2 • DMMddyy – Date format in MMDDYY • THHmmssS – Time format in THHmmssS.

Type	Example File Name	Description
Inbound	V_Q<<PackageUniqueID>>_Validation_Error.xml	esMD validation errors from esMD to RC for ICDT Request or Solicited or Unsolicited Response. • <<ReceiverRoutingId>> – RC Routing ID • Q – The Line of Business • V – Delivery type of the ICDT Solicited and Unsolicited Response • <<PackageUniqueID>> – 15-character RequestID or ResponseID • •
Outbound	<<ReceiverRoutingId>>.T.L<<CTC>>.U<<TransactionID>>.<<SenderRoutingID>>.DMMddyy.THHmmssS.zip	LETTERS, Decision Letters, and Pre-Pay and Post-Pay eMDR ADR letters from RC to esMD: • T – Environment ID. T is for UAT and P is for PROD • <<ReceiverRoutingId>> – RC Routing ID • L – The Line of Business • <<CTC>> – CTC of the program • U – Delivery type • <<TransactionID>> – 15-character esMD Transaction ID • <<SenderRoutingID>> – Sender Routing ID which is ESMD2 • DMMddyy – Date format in MMDDYY • THHmmssS – Time format in THHmmssS.
Outbound	<<ReceiverRoutingId>>.T.L<<CTC>>.W<<TransactionID>>.<<SenderRoutingID>>.DMMddyy.THHmmssS.zip	Post-Pay-Other eMDR ADR letters from RC to esMD: • T – Environment ID. T is for UAT and P is for PROD • <<ReceiverRoutingId>> – RC Routing ID • L – The Line of Business • <<CTC>> – CTC of the program • W – Delivery type • <<TransactionID>> – 15-character esMD Transaction ID • <<SenderRoutingID>> – Sender Routing ID which is ESMD2 • DMMddyy – Date format in MMDDYY THHmmssS – Time format in THHmmssS.

Type	Example File Name	Description
Inbound	<<ReceiverRoutingId>>.T.L<<CTC>>.E<<TransactionID>>.<<SenderRoutingID>>.DMMddyy.THHmmssS.zip	Service Registration flat file from esMD to RC: • T – Environment ID. T is for UAT and P is for PROD • <<ReceiverRoutingId>> – RC Routing ID • L – The Line of Business • <<CTC>> – CTC of the program which is “5” • E – Delivery type of the Service Registration Request • <<TransactionID>> – 15-character esMD Transaction ID • <<SenderRoutingID>> – Sender Routing ID which is ESMD2 • DMMddyy – Date format in MMDDYY • THHmmssS – Time format in THHmmssS.
Outbound	<<ReceiverRoutingId>>.T.L<<CTC>>.P<<TransactionID>>.<<SenderRoutingID>>.DMMddyy.THHmmssS.zip	Pickup notification file name from RC to esMD for the Service Registration: • T – Environment ID. T is for UAT and P is for PROD • <<ReceiverRoutingId>> – RC Routing ID • L – The Line of Business • <<CTC>> – CTC of the program which is “5” • P – Delivery type of the Pickup Notification for Service Registration Request • <<TransactionID>> – 15-character esMD Transaction ID • <<SenderRoutingID>> – Sender Routing ID which is ESMD2 • DMMddyy – Date format in MMDDYY • THHmmssS – Time format in THHmmssS.

Type	Example File Name	Description
Inbound	<<ReceiverRoutingId>>.T.L<<CTC>>.E<<TransactionID>>.<<SenderRoutingID>>.DMMddyy.THHmmssS.zip	Document Code flat file from esMD to RC: • T – Environment ID. T is for UAT and P is for PROD • <<ReceiverRoutingId>> – RC Routing ID • L – The Line of Business • <<CTC>> – CTC of the program which is “17” • E – Delivery type of the DCF Request • <<TransactionID>> – 15-character esMD Transaction ID • <<SenderRoutingID>> – Sender Routing ID which is ESMD2 • DMMddyy – Date format in MMDDYY THHmmssS – Time format in THHmmssS.
Outbound	<<ReceiverRoutingId>>.T.L<<CTC>>.P<<TransactionID>>.<<SenderRoutingID>>.DMMddyy.THHmmssS.zip	Pickup notification file name from RC to esMD for the document Code file: • T – Environment ID. T is for UAT and P is for PROD • <<ReceiverRoutingId>> – RC Routing ID • L – The Line of Business • <<CTC>> – CTC of the program which is “17” • P – Delivery type of the Pickup Notification for Document Code file • <<TransactionID>> – 15-character esMD Transaction ID • <<SenderRoutingID>> – Sender Routing ID which is ESMD2 • DMMddyy – Date format in MMDDYY THHmmssS – Time format in THHmmssS.

10. XML Schema Definitions

The following schema definitions are updated for the new format of the esMD Transaction ID:

1. esMD-businesstypes.xsd
2. esMD-config.xsd
3. esMD-rc.xsd
4. esMDProcessMetadata.xsd
5. emdr-rcprocessmetadata.xsd
6. emdr-postpay.xsd
7. emdr-postpay-other.xsd
8. letters_json_scema.json

11. XML Messages

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

11.1 Inbound

Note: Please refer to the Appendix A: Description of Fields on RC Client Tabs for details on how RC Client routes the inbound files once they are successfully processed into the data directories.

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

1. Payload Files in PDF and XML formats
2. Metadata File
3. Pickup HIH Status Response
4. Pickup Validation Error Response
5. Administrative Error HIH Status Response
6. Administrative Error Response Validation Error
7. PA Reject Response
8. PA Reject Validation Error Response
9. esMD Acknowledgement Response for the ADR Response/eMDR Request
10. esMD Validation Error Response for the ADR Response/eMDR Request
11. HIH Delivery Notification Response for the ADR Response/eMDR Request
12. ICDT Request XML
13. ICDT Solicited Response XML
14. ICDT Unsolicited Response XML
15. ICDT Pickup Notification/Acknowledgement Response (as a batch process)
16. ICDT Pickup Error Notification
17. ICDT Validation Error Notification
18. ICDT Acknowledgement Notification
19. ICDT Admin Error Response
20. Service Registration Request
21. HIH Delivery Notification for Service Registration Response
22. esMD Validation Error Response for Service Registration Response
23. esMD validation Error Response for Pre-Pay eMDR letters
24. esMD validation Error Response for Post-Pay eMDR letters
25. esMD validation Error Response for Post-Pay-Other eMDR letters
26. Document Code File
27. HIH Delivery Notification for LETTERS requests

11.1.1 Payload Files

The RC Client will receive payload files in the input directory with delivery type “E”. Examples of payload file names are E_JIT000000008418-AA569852P2545215519840365951551984040625_1.pdf or E_JIT000000008418-AA569852P2545215519840365951551984040625_2.xml

11.1.2 Metadata File

The metadata file accompanies the payload files as part of inbound documents sent to RC Client. These documents name will always start delivery type “E”, followed by content type code and global unique ID. The metadata file contains information about the payloads like the Object Identifier (OID), Transaction ID, Submission metadata (includes Attachment Control Number and other information), and optional metadata. The Content Type Code will change for each line of business. See Figure 22: E_L13_ BGR000007095735_metadata.xml.

Note: The metadata file will remain the same for all lines of business. For all list of line of business, please refer to Table 52: Content Type Code Descriptions

Note: The Claim ID is optional for First Level Appeal Requests and Second Level Appeal Requests.

Note: HIHs send new Claim ID updates for the acceptance of 8 numeric characters or the current ClaimId validations.

For more information on the Content Type Codes, refer to Appendix D: Content Type Codes.

As part of the October 2021 release, esMD included the specific PA program Content Type instead of Content Type 13. In addition, a metadata element was added to include the Workload Number in the metadata file.

Figure 22: E_L13_ BGR000007095735_metadata.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<ns0:RetrieveMedicalDocumentationResponse
xmlns:ns0="http://esmd.ois.cms.hhs.gov/v2/rc" returnCode="1"
serviceSuccessful="true">
    <statusDescription>The RetrieveMedicalDocumentationRequest processed
successfully.</statusDescription>
    <NumberOfDocuments>1</NumberOfDocuments>
    <ESMDPackage>
        <ESMDTransaction TransactionId="BGR000007095735" DeliveryType="E"/>
        <SendingOID>urn:oid:123.456.657.126</SendingOID>
        <TargetOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.1</TargetOID>
        <CompleteSubmission>true</CompleteSubmission>
        <RequestType>X12-XDR</RequestType>
        <SubmissionMetadata>
```

```
            <BusinessType>XDR X12</BusinessType>
            <CreationTime>2021-08-03T15:59:47.486-04:00</CreationTime>
            <SubmissionTime>2021-08-03T15:59:47.486-04:00</SubmissionTime>
            <EFTSubmissionTime>2021-08-03T15:59:47.486-
04:00</EFTSubmissionTime>
            <ContentTypeCode>8.5</ContentTypeCode>
            <WorkloadNumber>12302</WorkloadNumber>
            <NPI>1111111112</NPI>
        </SubmissionMetadata>
        <Documentation DocumentUniqueIdentifier="E_BGR000007095735-
SN125896P154323072930023451628020787049_0" MimeType="application/xml"
FileName="E_BGR000007095735-SN125896P154323072930023451628020787049_0.xml">
            <OptionalMetadata>
                <FieldName>FileName</FieldName>
                <FieldValue>E_BGR000007095735-
SN125896P154323072930023451628020787049_0.xml</FieldValue>
            </OptionalMetadata>
            <OptionalMetadata>
                <FieldName>Description</FieldName>
                <FieldValue>From esMD</FieldValue>
            </OptionalMetadata>
            <OptionalMetadata>
                <FieldName>CheckSum</FieldName>
                <FieldValue>
0b0e0016ff7e2b8c692e59c94f96777315aadc5362a14a7d00c454b72d3ed98f</FieldValue
>
            </OptionalMetadata>
            <OptionalMetadata>
                <FieldName>AttachmentControlNumber</FieldName>
                <FieldValue>SAKHHPCRsSPR34324</FieldValue>
            </OptionalMetadata>
        </Documentation>
    </ESMDPackage>
</ns0:RetrieveMedicalDocumentationResponse>
```

11.1.2.1 Split Payload Transactions

There is an optional functionality provided for HIHs to split the payloads when sending files are larger than 200 MB in size. Payloads that are larger than 200 MB in size are sent in multiple transactions by HIHs. In case of HIHs splitting the payloads when the sending files are larger than 200 MB in size, RCs will match/group the payloads using the additional information (ParentUniqueId and SplitNumber value set in the OptionalMetadata tag) in the RC metadata XML file. The same ParentUniqueId and a different SplitNumber (e.g., 1-5) value are passed in the RC Metadata XML file for all the transactions that are intended for a single submission by the HIH. RCs might receive duplicate split numbers or additional split numbers or missing split numbers for the same ParentUniqueID when HIHs are sending them.

Note: This is an optional functionality for HIHs to use when submitting payloads larger than 200 MB in size. Not all transactions received from HIHs have this additional information for RCs.

11.1.3 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. Once the esMD application receives the acknowledgement for the pickup notification from HIH, then it 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 Figure 23: N_L8_1_KBW000000006908_Delivery_Acknowledgement.xml.

Note: The metadata file will remain the same for all lines of business. For all list of line of business, please refer to Appendix D: Content Type Codes.

Figure 23: N_L8_1_KBW000000006908_Delivery_Acknowledgement.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<ns0:RCPickupNotificationResponse
xmlns:ns0="http://esmd.ois.cms.hhs.gov/v1/rc/config">
    <ESMDTransactionId>KBW000000006908</ESMDTransactionId>
    <ErrorInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:nil="true"/>
    <Status>Success</Status>
    <StatusDesc>SENT PICKUP STATUS TO HIH</StatusDesc>
</ns0:RCPickupNotificationResponse>
```

11.1.4 Pickup Validation Error Response

When the RC Client sends a Pickup Notification to esMD, the esMD application processes and sends the Pickup Notification to the HIH. If there is an error in processing the Pickup Notification submitted by the RC, the esMD application generates the Pickup Validation Error Response, as detailed in Figure 24: F_L13_PDW000000007903_Validation_Error.xml, and sends it to the RC. The RC will correct the pickup notification and resubmit it to esMD.

Figure 24: F_L13_PDW000000007903_Validation_Error.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<ns0:RCPickupNotificationResponse
xmlns:ns0="http://esmd.ois.cms.hhs.gov/v1/rc/config">
    <ESMDTransactionId>PDW000000007903</ESMDTransactionId>
    <ErrorInfo>
        <ErrorCode>520</ErrorCode>
        <ErrorName> </ErrorName>
        <ErrorDescription>Review Contractor Response Transaction ID does not
exist</ErrorDescription>
    </ErrorInfo>
    <Status>FAILED</Status>
```

```
    <StatusDesc>esMD validation error. Please correct and
resubmit.</StatusDesc>
    <fileName>ESMD2.V.L13. URAN012623160359.ES9996.D012623.T1603590.zip
</fileName>
</ns0:RCPickupNotificationResponse>
```

11.1.5 Administrative Error HIH Status Response

When the RC Client sends an administrative error for an inbound submission to esMD, the esMD application processes the administrative error and sends the response to the HIH. Once the esMD application receives the acknowledgement for the administrative error from HIH, then it generates the Administrative Error HIH Status Response and sends it to the RC, indicating the error was sent to the HIH, as detailed in the code in Figure 25: N_L1_IUC000000006217_Delivery_Acknowledgement.xml.

Note: The Administrative Error HIH Status Response will remain the same for all lines of business. For all list of line of business, please refer Table 52: Content Type Code Descriptions

Figure 25: N_L1_IUC000000006217_Delivery_Acknowledgement.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<ns0:SubmitPADeterminationResponseResult
xmlns:ns0="http://esmd.ois.cms.hhs.gov/v2/rc" returnCode="1"
serviceSuccessful="true">
    <statusDescription>Sent administrative error response delivery to
HIH</statusDescription>
    <ESMDTransaction TransactionId="IUC000000006217" DeliveryType="N"/>
</ns0:SubmitPADeterminationResponseResult>
```

11.1.6 Administrative Error Response Validation Error

When the RC Client sends an Administrative Error Response to esMD, the esMD application processes and sends the Administrative Error Response to the HIH. If there is an error in processing the Administrative Error Response submitted by the RC, the esMD application generates the Administrative Error Response Validation Error, as detailed in Figure 26: F_123456788912345_Validation_Error.xml, and sends it to the RC. The RC will correct the administrative error response and resubmits it.

Figure 26: F_123456788912345_Validation_Error.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<ns0:SubmitPADeterminationResponseResult
xmlns:ns0="http://esmd.ois.cms.hhs.gov/v2/rc" returnCode="1"
serviceSuccessful="true">
    <statusDescription>esMD validation error. Please correct and
resubmit.</statusDescription>
    <ESMDTransaction TransactionId="123456788912345" DeliveryType="F"
RoutingId="ESD002"/>

<ReceivedFileName>ESMD2.V.L1_6.URAN012623160359.ES9996.D012623.T1603590.zip
</ReceivedFileName>
    <ValidationFailure>
```

```
        <FailureCode>633</FailureCode>
        <FailureReason>esMD validation error: Either HIH is not active or
agreement has expired to receive the response.</FailureReason>
    </ValidationFailure>
    <ValidationFailure>
        <FailureCode>613</FailureCode>
        <FailureReason>esMD validation error : Administrative error code is
invalid. Correct and resubmit</FailureReason>
    </ValidationFailure>
</ns0:SubmitPADeterminationResponseResult>
```

11.1.7 PA Reject Validation Error Response

When the RC Client sends a PA Reject Response to esMD, the esMD application processes and sends the PA Reject Response to the HIH. If there is an error in processing the PA Reject Response submitted by the RC, the esMD application generates the PA Reject Response Error, as detailed in Figure 27: F_L9_QZF0011037065EC_Validation_Error.xml, and sends it to the RC. The RC will correct the response and resubmits the PA Reject Response.

Figure 27: F_L9_QZF0011037065EC_Validation_Error.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:SubmitPADeterminationResponseResult returnCode="0"
serviceSuccessful="false" xmlns:ns2="http://esmd.ois.cms.hhs.gov/v2/rc"
xmlns:ns3="http://esmd.ois.cms.hhs.gov/v2/rc/cmsbt">
    <statusDescription>ESMD VALIDATION ERROR. PLEASE CORRECT AND
RESUBMIT.</statusDescription>
    <ESMDTransaction TransactionId="QZF0011037065EC" DeliveryType="F"/>
    <ValidationFailure>
        <FailureCode>631</FailureCode>
        <FailureReason>esMD validation error: A Review or Error
Response is not allowed for this transaction.</FailureReason>
    </ValidationFailure>
    <ValidationFailure>
        <FailureCode>617</FailureCode>
        <FailureReason>esMD validation error: Mailbox ID in the
response does not match with the Mailbox ID that the request was
sent.</FailureReason>
    </ValidationFailure>
</ns2:SubmitPADeterminationResponseResult>
```

11.1.8 ICDT Request XML

The RCs send the ICDT Request to another RC via esMD in XML format as part of the ICDT Request Package with delivery type “Q”. The file name of the ICDT Request should contain only alphanumeric characters and underscores (i.e., “_”).

Figure 28: Q_QDMESD0020315191038490_ICDTSolicitedRequest.xml shows the XML message generated for an ICDT Request XML from RCs.

Figure 28: Q_QDMESD0020315191038490_ICDTSolicitedRequest.xml

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<ICDTRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://cms.hhs.gov/esmd/icdt">

<receiverOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.50</receiverOID>
    <receiverID>01232</receiverID>
    <senderOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.1</senderOID>
    <senderID>01231</senderID>
    <requestID>QDMESD0020315191038490</requestID>
    <contentType>15.1</contentType>
    <TransactionType transType="Claim">
        <OptionalMetadata>
            <FieldName>CLAIM_ID</FieldName>
            <FieldValue>12345678910</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>CASE_ID</FieldName>
            <FieldValue>12345678910</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>NPI</FieldName>
            <FieldValue>1234567890</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>HICN</FieldName>
            <FieldValue>1234567</FieldValue>
        </OptionalMetadata>
    </TransactionType>
</ICDTRequest>
```

11.1.9 ICDT Solicited Response XML

The RCs send the ICDT Response Document to another RC via esMD in any file format **except executable files** as part of the ICDT Response Package with delivery type “R”. The ICDT Solicited Response is sent for the Request from another RC. The file name of the ICDT Response Document should contain only alphanumeric characters and underscore (i.e., “_”).

Figure 29: R_RPXHES99960308191419170_ICDTSolicitedResponse.xml shows the sample XML Message of the ICDT Response sent from the RCs.

Figure 29: R_RPXHES99960308191419170_ICDTSolicitedResponse.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ICDTResponse xmlns="http://cms.hhs.gov/esmd/icdt">
    <receiverOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.4</receiverOID>
    <receiverID>01232</receiverID>
    <senderOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.3</senderOID>
    <senderID>01231</senderID>
    <requestID>QAC5K9XTESD0011210162133560</requestID>
    <contentType>15.2</contentType>
    <responseID>R7ESD0020130191302560</responseID>
    <TransactionType transType="Claim">
        <OptionalMetadata>
            <FieldName>CLAIM_ID</FieldName>
            <FieldValue>Claim ID 12345678910</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>CASE_ID</FieldName>
            <FieldValue>CASE ID 12345678910</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>NPI</FieldName>
            <FieldValue>1234567890</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>HICN</FieldName>
            <FieldValue>HICN123456</FieldValue>
        </OptionalMetadata>
    </TransactionType>
    <Documentation FileName="pdf-sample_1.pdf" MimeType="application/pdf"
DocUniqueID="pdf-sample_1">
        <OptionalMetadata>
            <FieldName>CheckSum</FieldName>
            <FieldValue>
0b0e0016ff7e2b8c692e59c94f96777315aadc5362a14a7d00c454b72d3ed98f</FieldValu
e>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>File Size</FieldName>
            <FieldValue>7945</FieldValue>
        </OptionalMetadata>
    </Documentation>
    <numberOfDocuments>1</numberOfDocuments>
</ICDTResponse>
```

11.1.10 ICDT Unsolicited Response XML

The RCs send the ICDT Unsolicited Response Document to another RC via esMD in any file format **except executable files** as part of the ICDT Response Package with delivery type “O”. The file name of the ICDT Response Document should contain only alphanumeric characters and underscore (i.e., “_”).

Figure 30: R_L153OQQES99960308191_ICDTUnSolicitedResponse.xml shows the sample XML Message of the ICDT Unsolicited Response sent from the RCs.

Figure 30: R_L153OQQES99960308191_ICDTUnSolicitedResponse.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ICDTResponse xmlns="http://cms.hhs.gov/esmd/icdt">
    <receiverOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.4</receiverOID>
    <receiverID>01232</receiverID>
    <senderOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.3</senderOID>
    <senderID>01231</senderID>
    <contentType>15.3</contentType>
    <responseID>RPQQES99960308191418450</responseID>
    <TransactionType transType="SMRC-Misroute">
        <OptionalMetadata>
            <FieldName>CLAIM_ID</FieldName>
            <FieldValue>Claim ID 12345678910</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>CASE_ID</FieldName>
            <FieldValue>CASE ID 12345678910</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>NPI</FieldName>
            <FieldValue>1234567890</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>HICN</FieldName>
            <FieldValue>HICN12345</FieldValue>
        </OptionalMetadata>
    </TransactionType>
    <Documentation FileName="pdf-sample_1.pdf" MimeType="application/pdf"
DocUniqueID="pdf-sample_1">
        <OptionalMetadata>
            <FieldName>CheckSum</FieldName>
            <FieldValue>
0b0e0016ff7e2b8c692e59c94f96777315aadc5362a14a7d00c454b72d3ed98f</FieldValue
>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>File Size</FieldName>
            <FieldValue>7945</FieldValue>
        </OptionalMetadata>
    </Documentation>
    <Documentation FileName="pdf-sample_2.pdf" MimeType="application/pdf"
DocUniqueID="pdf-sample_2">
        <OptionalMetadata>
            <FieldName>CheckSum</FieldName>
            <FieldValue>
0b0e0016ff7e2b8c692e59c94f96777315aadc5362a14a7d00c454b72d3ed98f</FieldValue
>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>File Size</FieldName>
            <FieldValue>7945</FieldValue>
        </OptionalMetadata>
    </Documentation>
```

```
    <numberOfDocuments>2</numberOfDocuments>
</ICDTResponse>
```

11.1.11 ICDT Pickup Notification/Acknowledgement Response

When the RC Client sends an ICDT Request or ICDT Solicited/Unsolicited Response to esMD, the esMD application validates the ICDT Request or ICDT Solicited/Unsolicited Response and generates the acknowledgement response or validation error and sends it to the RC using the Status API. The Recipient RC downloads the package and sends the successful pickup notification to esMD in real time using the Notification API. If there is any error in the Pickup Notification esMD sends the pickup error back to Recipient RC in real time. The pickup notification/acknowledgement response is generated and stored in esMD. All pickup notification/acknowledgements and validation errors are sent to RCs using the Status API which runs at scheduled intervals of time. The notifications and acknowledgments from esMD are not delivered to the RC in real time. All messages transmitted between RC and esMD are in JSON format. They are converted into XML and saved to appropriate ICDT user folders.

Note : Please refer ICD document for the ICDT JSON structures.

The RequestType element in the XML indicates whether it is a notification or acknowledgment for the particular transaction.

The RC Client Status API generates the acknowledgement response XML as shown in Figure 31: ICDT Acknowledgement Response.

Each request has an ICDT Notification block with all the details pertaining to the request. The number of requests in the XML file is identified by the ID value (highlighted).

Figure 31: ICDT Acknowledgement Response

```
<?xml version="1.0" encoding="UTF-8"?>
<ns0:ICDTCommunication xmlns:ns0="http://cms.hhs.gov/esmd/icdt">
    <ns0:ICDTNotification RequestType="SOLIC_REQ_ACK" id="1">
        <ns0:ICDTMetaData>

<ns0:receiverOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.50</ns0:receiverO
ID>
            <ns0:receiverID>01232</ns0:receiverID>

<ns0:senderOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.1</ns0:senderOID>
            <ns0:senderID>01231</ns0:senderID>
            <ns0:requestID>L151QDMESD0020315191</ns0:requestID>
            <ns0:contentType>15.1</ns0:contentType>
        </ns0:ICDTMetaData>
        <ns0:creationTime>2019-03-15T12:01:00.335-04:00</ns0:creationTime>
        <ns0:fileName>ESMD2.D.L15_1.
QWBT0002051601EC.ESD002.D031519.T1038490</ns0:fileName>
        <ns0:Status>
            <ns0:serviceSuccessful>true</ns0:serviceSuccessful>
```

```
        </ns0:Status>
    </ns0:ICDTNotification>
    <ns0:ICDTNotification RequestType="SOLIC_REQ_ACK" id="2">
        <ns0:ICDTMetaData>

<ns0:receiverOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.50</ns0:receiverO
ID>
            <ns0:receiverID>01232</ns0:receiverID>

<ns0:senderOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.1</ns0:senderOID>
            <ns0:senderID>01231</ns0:senderID>
            <ns0:requestID>L151QC2WESD002031519</ns0:requestID>
            <ns0:contentType>15.1</ns0:contentType>
        </ns0:ICDTMetaData>
        <ns0:creationTime>2019-03-15T12:01:00.335-04:00</ns0:creationTime>
        <ns0:fileName>ESMD2.D.L15_1.
QWBT0002051601EC.ESD002.D031519.T1038580.zip</ns0:fileName>
        <ns0:Status>
            <ns0:serviceSuccessful>true</ns0:serviceSuccessful>
        </ns0:Status>
    </ns0:ICDTNotification>
</ns0:ICDTCommunication>
```

11.1.12 ICDT Validation Error/Pickup Error Notification

When the RC Client sends an ICDT Request or ICDT Solicited/Unsolicited Response to esMD, the esMD application processes and sends the ICDT Request/Response to another RC. If there is an error processing the ICDT Request or ICDT Solicited/Unsolicited Response submitted by the RC at the esMD system, the esMD application generates the Validation Error Notification as detailed in Figure 32: Validation Error Response.xml and sends it to the RC using Status API.

Figure 32: Validation Error Response.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<ns0:ICDTCommunication xmlns:ns0="http://cms.hhs.gov/esmd/icdt">
    <ns0:ICDTNotificationFailure RequestType="RESP_VALDTN_ERR" id="1">
        <ns0:ICDTMetaData>

<ns0:receiverOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.50</ns0:receiverO
ID>
            <ns0:receiverID>01232</ns0:receiverID>

<ns0:senderOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.1</ns0:senderOID>
            <ns0:senderID>01231</ns0:senderID>
            <ns0:requestID> </ns0:requestID>
            <ns0:contentType>15.3</ns0:contentType>
        </ns0:ICDTMetaData>
        <ns0:creationTime>2019-03-15T11:25:18.457-04:00</ns0:creationTime>
        <ns0:responseID>L153RELESD0020315191</ns0:responseID>
        <ns0:fileName>D.ESMD2.D.L15_3.
RWBT0002051601ECRLEL.ESD002.D031519.T1040260.zip</ns0:fileName>
        <ns0:Status>
            <ns0:description>esMD validation error. Please correct and
resubmit.</ns0:description>
            <ns0:serviceSuccessful>false</ns0:serviceSuccessful>
```

```
        </ns0:Status>
        <ns0:ValidationFailure>
            <ns0:FailureCode>969</ns0:FailureCode>
            <ns0:FailureReason>esMD Validation Error: The documentation type
received in the ICDT UNSOLICITED RESPONSE XML is invalid. Correct and
Resubmit.</ns0:FailureReason>
        </ns0:ValidationFailure>
    </ns0:ICDTNotificationFailure>
</ns0:ICDTCommunication>
```

11.1.13 ICDT Administrative Error Response

The RCs send the following Administrative Error responses using the Status API for the ICDT Request/Solicited response and Unsolicited responses:

1. The file is corrupt and/or cannot be read.
2. When a virus/Infected file is detected, the RC Client logs the response JSON message and shuts down the RC Client application. No file is created in the error folder.

Figure 33: Administrative Error Response shows the sample Administrative error response JSON file.

Figure 33: Administrative Error Response

```
{
    "senderRoutingID": "ESMD2",
    "message": "ICDT Administrative Error Notification",
    "status": "FAILED",
    "ICDTNotifications": [
        {
            "statusDescription": "ERROR: Admin error notification",
            "filename":
"ES9998.D.L15_1.QEDB0003676201EC.ESMD2.D101222.T1313070.zip",
            "creationTime": "2023-04-03T10:15:14.566-04:00",
            "ICDTMetaData": {
                "senderID": "ES9999",
                "receiverID": "ESMD2",
                "receiverOID": "123.456.7890.12354",
                "requestID": "QUYTR9879650998UY",
                "contentType": "15.1",
                "senderOID": "098.876.6550.87655"
            },
            "esMDTransactionId": "EDB0003676201EC",
            "status": "FAILED",
            "errorDetails": [
                {
                    "errorCodeDescription": "File is corrupt and/or cannot
be read",
                    "errorCode": "631"
                }
            ]
        }
    ]
}
```

Figure 34: Virus Scan Error Response shows the sample Virus Scan error response JSON Message.

Figure 34: Virus Scan Error Response


```
{
    "senderRoutingID": "ESMD2",
    "message": "ICDT Administrative Error Notification",
    "status": "FAILED",
    "ICDTNotifications": [
        {
            "statusDescription": "ERROR: Admin error notification",
            "filename":
"ES9998.D.L15_1.QEDB0003676201EC.ESMD2.D101222.T1313070.zip",
            "creationTime": "2023-04-03T10:15:14.566-04:00",
            "ICDTMetaData": {
                "senderID": "ES9999",
                "receiverID": "ESMD2",
                "receiverOID": "123.456.7890.12354",
                "requestID": "QUYTR9879650998UY",
                "contentType": "15.1",
                "senderOID": "098.876.6550.87655"
            },
            "esMDTransactionId": "EDB0003676201EC",
            "status": "FAILED",
            "errorDetails": [
                {
                    "errorCodeDescription": "Submission is infected with a
virus.  This submission will not be processed by esMD.  Resubmit new
documentation.",
                    "errorCode": "ESMD_127"
                }
            ]
        }
    ]
}
```


11.1.14 esMD Validation Error Response for Pre-Pay eMDR Letters

When the RC Client sends a Pre-Pay eMDR package to esMD, and if there is an error processing the eMDR letters package submitted by the RC, the esMD application generates the Validation Error Response and sent to the RC using the Status API as detailed in Figure 35: esMD Validation Error Response for Pre-Pay eMDR Letters.

Figure 35: esMD Validation Error Response for Pre-Pay eMDR Letters

```
<?xml version="1.0" encoding="UTF-8"?>
<ns0:SubmitOutboundRequestOrResponseResult
xmlns:ns0="http://esmd.ois.cms.hhs.gov/v2/rc">
    <ESMDTransaction TransactionId="IYM000006820412" DeliveryType="F"
RoutingId="ESD002"/>
    <UniqueID>IYM000006820412</UniqueID>
    <submissionMetadata>
        <ContentTypeCode>2.5</ContentTypeCode>

<ReceivedFileName>D.ESMD2.D.L2_5.UWBT0002051601EC06B.ESD002.D040819.T1607070
.zip</ReceivedFileName>
    </submissionMetadata>
    <Status>
        <description>esMD validation error. Please correct and
resubmit.</description>
        <serviceSuccessful>true</serviceSuccessful>
    </Status>
    <ValidationFailure>
        <FailureCode>1041</FailureCode>
        <FailureReason>eMDR Process Metadata XML File is
missing.</FailureReason>
    </ValidationFailure>
</ns0:SubmitOutboundRequestOrResponseResult>
```

11.1.15 esMD Validation Error Response for Post-Pay/Post-Pay-Other eMDR Letters

When the RC Client sends a Post-Pay/Post-Pay-Other eMDR package to esMD, and if there is an error processing the eMDR letters package submitted by the RC, the esMD application generates the Validation Error Response. When there are validation errors in the eMDR Post-Pay/Post-Pay-Other structured XML, for the required elements, specific error message(s) will be sent with the name of the data element that failed the validation to allow the RC to correct any errors and resubmit the package as detailed in Figure 36: esMD Validation Error Response for Post-Pay/Post-Pay-Other eMDR Letters.

Figure 36: esMD Validation Error Response for Post-Pay/Post-Pay-Other eMDR Letters

```
<?xml version="1.0" encoding="UTF-8"?>
<ns0:SubmitOutboundRequestOrResponseResult
xmlns:ns0="http://esmd.ois.cms.hhs.gov/v2/rc">
    <ESMDTransaction TransactionId="EXJ000007095380" DeliveryType="F"
RoutingId="ESD002"/>
    <UniqueID>L16U1K6ES9996D072921T1241140</UniqueID>
    <submissionMetadata>
        <RCOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.1</RCOID>
        <RCName>Test Review Contractor</RCName>
        <ContentTypeCode>2.6</ContentTypeCode>
        <LetterId>gurramvt77kaojevyk6as3pyz8mk81uo</LetterId>
<!—The Delivery Type for Post-Pay-Other is W in the below file name -->
        <ReceivedFileName>ESMD2.D.L2_6.
UWBT0002051601EC.ESD002.D072921.T1241140.zip</ReceivedFileName>
    </submissionMetadata>
    <Status>
        <description>esMD validation error. Please correct and
resubmit.</description>
        <serviceSuccessful>true</serviceSuccessful>
    </Status>
    <ValidationFailure>
        <FailureCode>1082</FailureCode>
        <FailureReason>esMD Validation Error: EITHER THE PROVIDER CITY IS
INVALID OR MISSING</FailureReason>
    </ValidationFailure>
</ns0:SubmitOutboundRequestOrResponseResult>
```

11.1.16 Letters JSON Message

The RC Client uploads a Letters JSON message to esMD in real time. If there is an error processing the request submitted by the RC, the esMD application generates the Validation Error Response. When there are validation errors in the Letters JSON message, for the required elements, specific error message(s) will be sent with the name of the data element that failed the validation to allow the RC to correct any errors and resubmit the package as detailed in Letters request as shown in Figure 37: esMD Validation Error Response for Letters.

If there are no errors the request is sent to HIH and the Delivery Confirmation notification is generated and saved in the notification folder in real time as shown in Figure 38: esMD HIH Delivery Notification for .

Figure 37: esMD Validation Error Response for Letters


```
    "esmdtransactionid": "CHV0007151216EC",
    "routingId": "ESD002",
    "hihOid": null,
    "rcOid": "urn:oid:2.16.840.1.113883.13.34.110.1.999.1",
    "rcName": "Test Review Contractor",
    "letterId": "TestLetterId",
    "contentType": "20",
    "status": "FAILED",
    "statusDescription": "LETTERS_SCHEMA_000",
    "errorDetails": [
    {
        "errorcode": "\"linesofbusiness\" length must be less than or equal
to 10 characters long",
        "errormessage": "esMD Validation error:undefined"
    },
    {
        "errorcode": "LETTERS_009",
        "errormessage": "esMD Validation error: PROVIDER NPI IS INVALID IN
LETTERS"
    }
]
}
```


Figure 38: esMD HIH Delivery Notification for LETTERS


```
{
    "esmdtransactionid": "YPR0007151172EC",
    "routingId": "ESD002",
    "hihOid": "urn:oid:123.456.657.126",
    "rcOid": "urn:oid:2.16.840.1.113883.13.34.110.1.999.1",
    "rcName": "DATS",
    "letterId": "Asdwe",
    "contentType": "20",
    "status": "SUCCESS",
    "statusDescription": "Letters Processed and delivered to HIH.",
    "errorDetails": []
}
```


11.1.17 Document Code File

Document code file is a flat file sent by esMD on a Quarterly basis. Figure 39: Document Code File shows the sample Document Code File.

Figure 39: Document Code File

```
U202001311143015DOCUCODE

V0000001UBPDDJVThe long description of the document which is getting requested

V0000001UBPDDJVThe long description of the document which is getting requested

V0000001UBPDDJVThe long description of the document which is getting requested
```

V0000001UBPDDJVThe long description of the document which is getting requested

V0000001UBPDDJVThe long description of the document which is getting requested

V0000001UBPDDJVThe long description of the document which is getting requested

V0000001UBPDDJVThe long description of the document which is getting requested

W0000007

11.2 Outbound

Note for API users: Please refer to the properties files in the resources folder packaged with the source code for more details on the reference data needed to populate the outbound XMLs described in this section.

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

1. Pickup Notification
2. Error Pickup Notification
3. Review Decision Response to PA Request
4. Error Response to PA request
5. Administrative Error Response to Inbound Submissions
6. esMD Process Metadata (eMDR Request)
7. ADR Review Response XML
8. esMD Process Metadata (ADR Review Response)
9. ICDT Request
10. ICDT Solicited Response
11. ICDT Unsolicited Response
12. ICDT Pickup/Pickup Error Notification
13. ICDT Administrative Error Notification
14. Pickup Notification for Service Registration
15. Pickup Notification for Document Codes
16. eMDR Process Metadata
17. eMDR Structured File for Post-Pay ADR letters
18. eMDR Structured File for Post-Pay-Other ADR letters
19. API Error Messages for Pre-Pay, Post-Pay and Post-Pay-Other
20. DCF Pickup Notification
21. DCF Error Pickup Notification
22. HOPD Pickup Notification
23. LETTERS Request.

11.2.1 Pickup Notification

The RC Client generates pickup notifications for all inbound files with delivery type “E” pulled that are from esMD and processed successfully as detailed Figure 40: Pickup Notification.

Note: ESMDTransactionId is mandatory in Pickup Notification. The Pickup Notifications are sent to esMD in real time in JSON format.

Figure 40: Pickup Notification

```
{
    "notification": [
        {
            "pickupTime": "2023-03-30T13:29:30.6910000-04:00",
            "filename":
"PT9995.V.L1.EDKQ0011105965EC.ESMD2.D033023.T1327480.zip",
            "esMDTransactionId": "DKQ0011105965EC",
            "submissionTime": "2023-03-30T13:29:30.6910000-04:00",
            "status": "SUCCESS",
            "errorMessages": [

            ]
        }
    ],
    "senderRoutingId": "PT9995",
    "notificationType": "PICKUP"
}
```

11.2.2 Error Pickup Notification

The RC Client generates pickup error notifications for all inbound files pulled from esMD Cloud and processed unsuccessfully, as detailed in Figure 41: Pickup Error Notification JSON File. The processing errors are generated in two scenarios.

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

Figure 41: Pickup Error Notification JSON File

```
{
    "notification": [
        {
            "pickupTime": "2023-03-30T13:29:30.6910000-04:00",
            "filename":
"PT9995.V.L1.EDKQ0011105965EC.ESMD2.D033023.T1327480.zip",
            "esMDTransactionId": "DKQ0011105965EC",
            "submissionTime": "2023-03-30T13:29:30.6910000-04:00",
            "status": "FAILED",
            "errorMessages": [
                {
```

```
                    "errorName": "ERROR EXTRACTING DOCUMENT",
                    "errorDescription": "ESMD_534 - RC Client processing
error (Unzip failure).  Please resubmit.",
                    "errorCode": "534"
                }
            ],

        }
    ],
    "senderRoutingId": "PT9995",
    "notificationType": "PICKUP"
}
```


11.2.3 Error Response to PA Request

The Error Response to PA Request is the XML message from the RC to the HIH, to inform the HIH of the review result response with decision as “Rejected” as detailed in Figure 42: PA Reject Error Response.

As part of the April 2025 release, esMD will start accepting a new error code in services in the PA Reject Response for any XDR HHPCR program. When the procedure code is repeated in the same billing period (error codes 57 and 15), RCs need to send a response with error code 57 for erroneous records and either an empty value, or any error code from the service level for non-erroneous records.

Please refer to the Appendix B:Reject Error Codes for more information on the error codes used in the Error Review Response for a PA Request. The PA Reject Error messages are saved in the output/notifications folder in the data directory in JSON format.

Figure 42: PA Reject Error Response


```
{
    "senderRoutingId": "ES9997",
    "notificationType": "PAREJECT",
    "paResponses": [
        {
            "creationTime": "2023-03-31T14:06:12.7340000-04:00",
            "deliveryType": "R",
            "submissionTime": "2023-03-31T14:06:48.0280000-04:00",
            "esMDTransactionId": "QZF0011037065EC",
            "errorResponseDetails": [
                {
                    "rejectErrorCodeRecords": [
                        {
                            "errorCategoryName": "Facility",
                            "errorCodeRecords": [
                                {
                                    "errorCodeDescription": " Provider
address is missing or invalid",
                                    "errorCode": "97"
                                },
                                {
```

```
                                    "errorCodeDescription": "NPI is missing
or invalid",
                                    "errorCode": "51"
                                }
                            ]
                        }
                    ],
                    "reasonCodeList": [
                        "12345",
                        "56788"
                    ],
                    "decisionIndicator": "R",
                    "requestLevelUniqueTrackingNumber": "987464590"
                }
            ]
        }
    ]
}
```


11.2.4 Administrative Error Response to Inbound Submissions

The Administrative Error Response is the JSON message from the RC to the HIH to inform the HIH of the administrative error response to inbound submissions.

Note: Section 5 How to Submit an Inbound Submission Error on the Administrative Error Response to Inbound Submissions Tab describes the process of creating an JSON message, using the RC Client.

Figure 43: EPP000000008983_adminerror.json provides the PA Review Decision Response.

Figure 43: EPP000000008983_adminerror.json


```
    ],
            "filename": "",
            "creationTime": "2023-03-31T13:43:47.6080000-04:00",
            "contenttypecd": "",
            "esMDTransactionId": "VME0004332201EC",
            "submissionTime": "2023-03-31T13:43:47.6150000-04:00",
            "status": "Success"
        }
    ],
    "senderRoutingId": "ES9997",
    "notificationType": "ADMINERROR"
}
```


11.2.5 ICDT Request

The RCs send the ICDT Request to another RC via esMD in XML format as part of the ICDT Request Package with delivery type “Q”. The file name of the ICDT Request should contain only alphanumeric characters and underscores (i.e., “_”).

Figure 44: Q_QDME091622031519_ICDTSolicitedRequest.xml shows the XML message generated for an ICDT Request XML from the RCs.

Figure 44: Q_QDME091622031519_ICDTSolicitedRequest.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ICDTRequest xmlns="http://cms.hhs.gov/esmd/icdt">
    <receiverOID>urn:oid:126.543.321.123</receiverOID>
    <receiverID>01232</receiverID>
    <senderOID>urn:oid:126.543.321.121</senderOID>
    <senderID>01231</senderID>
    <requestID> L1518DMESD0020315191038490</requestID>
    <contentType>15.1</contentType>
    <TransactionType transType="Claim">
        <OptionalMetadata>
            <FieldName>CLAIM_ID</FieldName>
            <FieldValue>Claim ID 12345678910</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>CASE_ID</FieldName>
            <FieldValue>CASE ID 12345678910</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>NPI</FieldName>
            <FieldValue>1234567890</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>HICN</FieldName>
            <FieldValue>HICN12345678910</FieldValue>
        </OptionalMetadata>
    </TransactionType>
</ICDTRequest>
```

11.2.6 ICDT Solicited Response

The RCs send the ICDT Response Document to another RC via esMD in any file format **except executable files** as part of the ICDT Response Package with delivery type “R”. The ICDT Solicited Response is sent for the Request from another RC. The file name of the ICDT Response Document should contain only alphanumeric characters and underscores (i.e., “_”).

Figure 45: R_RDME091622031519_ICDTSolicitedResponse.xml shows the sample XML Message of the ICDT Response sent from the RCs.

Figure 45: R_RDME091622031519_ICDTSolicitedResponse.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ICDTResponse xmlns="http://cms.hhs.gov/esmd/icdt">
    <receiverOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.4</receiverOID>
    <receiverID>01232</receiverID>
    <senderOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.3</senderOID>
    <senderID>01231</senderID>
```

```
    <requestID>QAC5K9XTESD0011210162133560</requestID>
    <contentType>15.2</contentType>
    <responseID>L1521R7ESD0020130191302560</responseID>
    <TransactionType transType="Claim">
        <OptionalMetadata>
            <FieldName>CLAIM_ID</FieldName>
            <FieldValue>Claim ID 12345678910</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>CASE_ID</FieldName>
            <FieldValue>CASE ID 12345678910</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>NPI</FieldName>
            <FieldValue>1234567890</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>HICN</FieldName>
            <FieldValue>HICN123456</FieldValue>
        </OptionalMetadata>
    </TransactionType>
    <Documentation FileName="pdf-sample_1.pdf" MimeType="application/pdf"
DocUniqueID="pdf-sample_1">
        <OptionalMetadata>
            <FieldName>CheckSum</FieldName>
            <FieldValue>
0b0e0016ff7e2b8c692e59c94f96777315aadc5362a14a7d00c454b72d3ed98f</FieldValue
>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>File Size</FieldName>
            <FieldValue>7945</FieldValue>
        </OptionalMetadata>
    </Documentation>
    <numberOfDocuments>1</numberOfDocuments>
</ICDTResponse>
```

11.2.7 ICDT Unsolicited Response

The RCs send the ICDT Unsolicited Response Document to another RC via esMD in any file format **except executable files** as part of the ICDT Response Package with delivery type “R”. The file name of the ICDT Response Document should contain only alphanumeric characters and underscores (i.e., “_”).

Figure 46: R_RDME091622031519_ICDTUnSolicitedResponse.xml shows the sample XML Message of the ICDT Unsolicited Response sent from the RCs.

Figure 46: R_RDME091622031519_ICDTUnSolicitedResponse.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ICDTResponse xmlns="http://cms.hhs.gov/esmd/icdt">
    <receiverOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.4</receiverOID>
    <receiverID>01232</receiverID>
    <senderOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.3</senderOID>
```

```
    <senderID>01231</senderID>
    <contentType>15.3</contentType>
    <responseID>L1535RJES99960308191422450</responseID>
    <TransactionType transType="SMRC-Misroute">
        <OptionalMetadata>
            <FieldName>CLAIM_ID</FieldName>
            <FieldValue>Claim ID 12345678910</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>CASE_ID</FieldName>
            <FieldValue>CASE ID 12345678910</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>NPI</FieldName>
            <FieldValue>1234567890</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>HICN</FieldName>
            <FieldValue>HICN12345</FieldValue>
        </OptionalMetadata>
    </TransactionType>
    <Documentation FileName="pdf-sample_1.pdf" MimeType="application/pdf"
DocUniqueID="pdf-sample_1">
        <OptionalMetadata>
            <FieldName>CheckSum</FieldName>
            <FieldValue>
0b0e0016ff7e2b8c692e59c94f96777315aadc5362a14a7d00c454b72d3ed98f</FieldValue
>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>File Size</FieldName>
            <FieldValue>7945</FieldValue>
        </OptionalMetadata>
    </Documentation>
    <Documentation FileName="pdf-sample_2.pdf" MimeType="application/pdf"
DocUniqueID="pdf-sample_2">
        <OptionalMetadata>
            <FieldName>CheckSum</FieldName>
            <FieldValue>
0b0e0016ff7e2b8c692e59c94f96777315aadc5362a14a7d00c454b72d3ed98f</FieldValue
>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>File Size</FieldName>
            <FieldValue>7945</FieldValue>
        </OptionalMetadata>
    </Documentation>
    <numberOfDocuments>2</numberOfDocuments>
</ICDTResponse>
```

11.2.8 ICDT Pickup/Pickup Error Notification

When the RC Client sends an ICDT Request or ICDT Response to esMD, the esMD application processes the response and sends the acknowledgement response to the RC after successfully validating the response in the esMD system. The Recipient RC downloads the package and sends the successful pickup notification to esMD. The Acknowledgement Responses are sent to RCs using the Status API.

The esMD system generates the acknowledgement response to RC as shown in Figure 47: ICDT Acknowledgement Response and sends it to the RC.

Figure 47: ICDT Acknowledgement Response

```
<?xml version="1.0" encoding="UTF-8"?>
<ns0:ICDTCommunication xmlns:ns0="http://cms.hhs.gov/esmd/icdt">
    <ns0:ICDTNotification RequestType="SOLIC_REQ_ACK" id="1">
        <ns0:ICDTMetaData>

<ns0:receiverOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.50</ns0:receiverO
ID>
            <ns0:receiverID>01232</ns0:receiverID>

<ns0:senderOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.1</ns0:senderOID>
            <ns0:senderID>01231</ns0:senderID>
            <ns0:requestID>L1518DMESD0020315191038490</ns0:requestID>
            <ns0:contentType>15.1</ns0:contentType>
        </ns0:ICDTMetaData>
        <ns0:creationTime>2019-03-15T12:01:00.335-04:00</ns0:creationTime>
        <ns0:fileName>ESMD2.D.L15_1.
QWBT0002051601EC.ESD002.D031519.T1038490</ns0:fileName>
        <ns0:Status>
            <ns0:serviceSuccessful>true</ns0:serviceSuccessful>
        </ns0:Status>
    </ns0:ICDTNotification>
    <ns0:ICDTNotification RequestType="SOLIC_REQ_ACK" id="2">
        <ns0:ICDTMetaData>

<ns0:receiverOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.50</ns0:receiverO
ID>
            <ns0:receiverID>01232</ns0:receiverID>

<ns0:senderOID>urn:oid:2.16.840.1.113883.13.34.110.1.999.1</ns0:senderOID>
            <ns0:senderID>01231</ns0:senderID>
            <ns0:requestID>L151C2WESD0020315191038580</ns0:requestID>
            <ns0:contentType>15.1</ns0:contentType>
        </ns0:ICDTMetaData>
        <ns0:creationTime>2019-03-15T12:01:00.335-04:00</ns0:creationTime>
        <ns0:fileName>ESMD2.D.L15_1.
QWBT0002051601EC.ESD002.D031519.T1038580</ns0:fileName>
        <ns0:Status>
            <ns0:serviceSuccessful>true</ns0:serviceSuccessful>
        </ns0:Status>
    </ns0:ICDTNotification>
</ns0:ICDTCommunication>
```

11.2.9 Service Registration Pickup Notification

After picking up the Service Registration Request batch file, the RC generates the pickup notification and sends to esMD. Refer to Figure 48: Pickup Notification for sample pickup notification.

Figure 48: Pickup Notification


```
{
    "notification": [
        {
            "pickupTime": "2023-03-30T13:29:30.6910000-04:00",
            "filename":
"PT9995.V.L5.EDKQ0011105965EC.ESMD2.D033023.T1327480.zip",
            "esMDTransactionId": "DKQ0011105965EC",
            "submissionTime": "2023-03-30T13:29:30.6910000-04:00",
            "status": "SUCCESS"
        },
                "errorMessages": [

            ]
    ],
    "senderRoutingId": "PT9995",
    "notificationType": "PICKUP"
}
```


11.2.10 eMDR Process Metadata (Pre-Pay eMDR letters)

The eMDR Process metadata file accompanies the ADR letters zip files for Pre-Pay functionality as the outbound document package with the delivery type “U”.

A new additional NPI metadata element is added at the Documentation level (individual PDF Letter level) in the process metadata XML when creating the eMDR prepay zip files.

The Content Type Code for the eMDR Pre-Pay is changed from 1.5 to 2.5.

No other changes are intended except for the new metadata element to populate the NPI value in the eMDR Process Metadata File.

Below are the new NPI metadata validations performed in the RC Client application when creating the prepay eMDR zip file.

1. The RC Client API Pre-Pay process validates if the NPI is present in the eMDR Process metadata file when creating the Pre-Pay zip file.
2. The RC Client API prepay process validates the NPI length and format in the eMDR Process Metadata file. The field name NPI is not case sensitive.
3. The NPI value should be numeric and 10 digits in length. If there are multiple PDF files received in the zip package, a NPI should be present for each PDF present in the package under the Documentation section.

Updated the unique ID format from 15 random alphanumeric characters to 3 random characters and appended with datetime stamp.

Format: <<DeliveryType>><<3 Random Characters>><< MMddyyHHmmss>>

The sample XML files are included in Figure 49: Sample Pre-Pay eMDRProcessMetadata XML. The same eMDR Process Metadata schema is used for Post-Pay/Post-Pay-Other functionality.

The PDF Filename must be created with the below file naming convention. The length of the filename should not exceed 60 characters. The pdf filename validation is not performed in the RC Client application. LetterId is random alphanumeric characters and Datetime can be in any numeric format.

PDF Filename Format: <<LetterId>>_<<Datetime>>_<<Random Alphanumeric>>.pdf

Example: TESTLETTERID5678_20190405_REVIEW1.pdf

Figure 49: Sample Pre-Pay eMDRProcessMetadata XML

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<eMDRProcessMetadata
xmlns="http://esmd.ois.cms.hhs.gov/rc/v1/emdr/processmetadata">
    <uniqueID>UY6K010224090200</uniqueID>
    <numberOfDocuments>38</numberOfDocuments>
    <submissionMetadata>
        <creationTime>2023-10-18T17:00:47.5710000-04:00</creationTime>
        <routingName>ES0015</routingName>
        <deliveryType>U</deliveryType>
        <contentTypeCode>2.5</contentTypeCode>
    </submissionMetadata>
    <Documentation
DocumentUniqueIdentifier="1300122327800963304CTA0PR_20231017235959_L075ESMD"
MimeType="application/pdf"
FileName="1300122327800963304CTA0PR_20231017235959_L075ESMD.pdf">
        <OptionalMetadata>
            <FieldName>CheckSum</FieldName>
            <FieldValue>
0b0e0016ff7e2b8c692e59c94f96777315aadc5362a14a7d00c454b72d3ed98f</FieldValue
>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>File Size</FieldName>
            <FieldValue>50053</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>NPI</FieldName>
            <FieldValue>1243878901</FieldValue>
        </OptionalMetadata>
    </Documentation>
    <Documentation
DocumentUniqueIdentifier="1300122327800962104CTA0PR_20231017235959_L075ESMD"
MimeType="application/pdf"
FileName="1300122327800962104CTA0PR_20231017235959_L075ESMD.pdf">
```

```
        <OptionalMetadata>
            <FieldName>CheckSum</FieldName>
            <FieldValue>
0b0e0016ff7e2b8c692e59c94f96777315aadc5362a14a7d00c454b72d3ed98f</FieldValue
>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>File Size</FieldName>
            <FieldValue>49987</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>NPI</FieldName>
            <FieldValue>9876548394</FieldValue>
        </OptionalMetadata>
    </Documentation>
    <Documentation
DocumentUniqueIdentifier="13001223278009600004CTA0PR_20231017235959_L075ESMD"
MimeType="application/pdf"
FileName="13001223278009600004CTA0PR_20231017235959_L075ESMD.pdf">
        <OptionalMetadata>
            <FieldName>CheckSum</FieldName>
            <FieldValue>
0b0e0016ff7e2b8c692e59c94f96777315aadc5362a14a7d00c454b72d3ed98f</FieldValue
>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>File Size</FieldName>
            <FieldValue>50048</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>NPI</FieldName>
            <FieldValue>0876484989</FieldValue>
        </OptionalMetadata>
    </Documentation>
</eMDRProcessMetadata>
```

11.2.11 eMDR Process Metadata (Post-Pay/Post-Pay-Other eMDR letters)

The eMDR Process metadata file accompanies the ADR letters zip files for Pre-Pay functionality as the outbound document package with the delivery type “U” for Post-Pay and “W” for Post-Pay-Other.

The Content Type Code for the eMDR Post-Pay and Post-Pay-Other is changed from 1.6 to 2.6.

The sample XML files are included in Figure 50: Sample Post-Pay/Post-Pay-Other eMDRProcessMetadata XML. The same eMDR Process Metadata schema is used for Pre-Pay functionality.

Figure 50: Sample Post-Pay/Post-Pay-Other eMDRProcessMetadata XML

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<eMDRProcessMetadata
xmlns="http://esmd.ois.cms.hhs.gov/rc/v1/emdr/processmetadata">
    <uniqueID>URUM010224110400</uniqueID>
<!--Format: <<DelvieryType>><<3 Random Characters>><< MMddyyHHmmss>> -- >
    <numberOfDocuments>3</numberOfDocuments>
    <submissionMetadata>
        <creationTime>2019-05-02T13:56:20.561-04:00</creationTime>
        <routingName>ESD002</routingName>
<!-- DeliveryType is 'W' for postpay-other  -->

        <deliveryType>U</deliveryType>
        <contentTypeCode>2.6</contentTypeCode>
    </submissionMetadata>
    <Documentation
DocumentUniqueIdentifier="TESTLETTERID4478_20190405&amp;_REVIEW3"
MimeType="application/pdf"
FileName="TESTLETTERID4478_20190405&amp;_REVIEW3.pdf">
        <OptionalMetadata>
            <FieldName>CheckSum</FieldName>

<FieldValue>8c84d07d505ad1429725e5f09a79daf96465592d</FieldValue>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>File Size</FieldName>
            <FieldValue>105255200</FieldValue>
        </OptionalMetadata>
    </Documentation>
    <Documentation
DocumentUniqueIdentifier="U_UGUI185831202209_eMDRStructuredFile"
MimeType="application/xml"
FileName="U_UGUI185831202209_eMDRStructuredFile.xml">
        <OptionalMetadata>
            <FieldName>CheckSum</FieldName>
            <FieldValue>
0b0e0016ff7e2b8c692e59c94f96777315aadc5362a14a7d00c454b72d3ed98f</FieldValue
>
        </OptionalMetadata>
        <OptionalMetadata>
            <FieldName>File Size</FieldName>
            <FieldValue>2074</FieldValue>
        </OptionalMetadata>
    </Documentation>
</eMDRProcessMetadata>
```

11.2.12 eMDR Structured File for Post-Pay ADR letters

Figure 51: U_U16Y072622210346_eMDRStructuredFile.xml shows the sample eMDR structured ADR letter file for Post-Pay eMDR letters.

Figure 51: U_U16Y072622210346_eMDRStructuredFile.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<EMDRPostPayRequest
xmlns="http://esmd.ois.cms.hhs.gov/v3/rc/esmd/emdr/postpay">
    <eMDRType>POST-PAY</eMDRType>
    <uniqueLetterId>Letter2</uniqueLetterId>
```

```
    <letterDate>2019-03-13</letterDate>
    <respondTo>
        <organizationName>Keebler, Halvorson and Murphy</organizationName>
        <addressLine1>28569 Conn Manors</addressLine1>
        <city>Darionland</city>
        <state>RI</state>
        <zipCode>193084149</zipCode>
    </respondTo>
    <senderDetails>
        <organizationName>Marvin, Gleason and Hermiston</organizationName>
    </senderDetails>
    <providerDetails>
        <organizationOrlastName>Cartwright</organizationOrlastName>
        <addressLine1>4240 Isabel Ports</addressLine1>
        <city>South Janelle</city>
        <state>ME</state>
        <zipCode>410257281</zipCode>
        <npi>1932695756</npi>
    </providerDetails>
    <letterDetails>
        <respondBy>2019-09-04</respondBy>
        <jurisdiction>DW</jurisdiction>
        <programName>PARTB</programName>
    </letterDetails>
    <reviewLevelRecordList>
        <reviewLevel>
            <analysisRecordList>
                <analysisRecord>
                    <analysisID>EKNSUKTAKTSHMSQNVBDA</analysisID>
                    <claimLevelItemList>
                        <claimSetLevel>
                            <claimDetails>
                                <claimID>BCQWQLZNJXKFFQUMVEAF</claimID>
                                <beneficiaryID>BKSKIKK</beneficiaryID>

<beneficiaryLastName>Bogan</beneficiaryLastName>
                            </claimDetails>
                        </claimSetLevel>
                    </claimLevelItemList>
                </analysisRecord>
            </analysisRecordList>
        </reviewLevel>
    </reviewLevelRecordList>
</EMDRPostPayRequest>
```

11.2.13 eMDR Structured File for Post-Pay-Other ADR letters

Figure 52: W_L16Y5XESD0020726192103460_eMDRStructuredFile.xml shows the sample eMDR structured ADR letter file for Post-Pay-Other eMDR letters.

Figure 52: W_L16Y5XESD0020726192103460_eMDRStructuredFile.xml

```
<?xml version="1.0" encoding="utf-8"?>
<EMDRPostPay2Request>
    <adrLevelInfo>
```

```
        <eMDRType>POST-PAY-OTHER</eMDRType>
        <uniqueLetterId>U6556565665SM17668688</uniqueLetterId>
        <letterDate>11/16/2022</letterDate>
    </adrLevelInfo>
    <rcSystemIdentifierDetails>
        <rcSystemIdentifier>RC765326623</rcSystemIdentifier>
        <medicareAppeal>1152578678</medicareAppeal>
        <caseNumber>UCM2376674</caseNumber>
    </rcSystemIdentifierDetails>
    <senderDetails>
        <organizationName>Keebler, Halvorson and Murphy</organizationName>
        <addressLine1>28569 Conn Manors</addressLine1>
        <addressLine2>Quail valley</addressLine2>
        <city>Darionland</city>
        <state>RI</state>
        <zipCode>19308-4149</zipCode>
        <telephonewithExt>8887797477-5636</telephonewithExt>
        <emailAddress>CERTprovider@nciinc.com</emailAddress>
        <webSiteAddress>https://www.cms.gov/</webSiteAddress>
    </senderDetails>
    <providerDetails>
        <lastNameOrOrganizationName>Cartwright</lastNameOrOrganizationName>
        <npi>1932695756</npi>
        <firstName>Michael GWYNN</firstName>
        <middleName>PA, Co</middleName>
        <addressLine1>4240 Isabel Ports</addressLine1>
        <addressLine2>Hidden valley</addressLine2>
        <city>South Janelle</city>
        <state>ME</state>
        <zipCode>41025-7281</zipCode>
        <providerNumberOrPTAN>5443355343</providerNumberOrPTAN>
        <fax>8042618100</fax>
    </providerDetails>
    <letterDetails>
        <respondBy>11/16/2022</respondBy>
        <jurisdiction>UPIC Mid-Western</jurisdiction>
        <programName>PARTB</programName>
        <letterSequence>First</letterSequence>
        <previousLetterDate>11/16/2022</previousLetterDate>
        <appealCrossRef>RC7665665777</appealCrossRef>
        <redetermination>Appeal 55427-544552</redetermination>
        <reconsideration>Appeal 564546</reconsideration>
    </letterDetails>
    <submissionContactsList>
        <submissionContacts>
            <contactName>>Wiza - Harber</contactName>
            <contactTelephone>7666666666</contactTelephone>
            <contactFax>987766666</contactFax>
            <contactEmail>Test1@CMS.gov</contactEmail>
        </submissionContacts>
    </submissionContactsList>
    <reviewLevelDetailsList>
        <reviewLevelDetail>
            <documentCodeList>
                <documentCode>100001800001700003</documentCode>
                <documentCode>100573</documentCode>
```

```
                <documentCode>100574</documentCode>
                <documentCode>100575</documentCode>
                <documentCode>100576</documentCode>
                <documentCode>100577</documentCode>
                <documentCode>100578</documentCode>
                <documentCode>100579</documentCode>
            </documentCodeList>
            <inquiryText1>A_123333</inquiryText1>
            <inquiryText2>A1_223333</inquiryText2>
            <inquiryText3>B1_7663653662</inquiryText3>
            <inquiryText4>B2_676456627673</inquiryText4>
        </reviewLevelDetail>
    </reviewLevelDetailsList>
    <claimDetailsList>
        <claimDetail>
            <beneficiaryID>BKSKIKK</beneficiaryID>
            <claimID>BCQWQLZNJXKFFQUMVEAF</claimID>
            <fromDateOfService>11/10/2022</fromDateOfService>
            <toDateOfService>11/16/2022</toDateOfService>
        </claimDetail>
    </claimDetailsList>
</EMDRPostPay2Request>
```

11.2.14 API Error Messages for Pre-Pay, Post-Pay, and Post-Pay-Other

Table 10: RC Client Error Codes and Error Messages lists the validation error messages while generating the Pre-Pay, Post-Pay and Post-Pay-Other zip packages in the RC Client API.

Table 10: RC Client Error Codes and Error Messages

ID	Scenario	Error Code	Error Message
1	Attachments other than PDF format for Pre-Pay packages	ADR_ESMD_LETTERS_FILE_INVALID_ATTACHMENT	ADR esMD Letters File must be in PDF format
2	File size is 0 MB or greater than 140 MB in size for Pre-Pay packages	ADR_ESMD_LETTERS_FILE_EXCEEDS_MAX_LIMIT	ADR esMD PDF Letters exceeds 140 MB
3	Missing ADR letter file	MISSING_ADR_ESMD_LETTERS_FILE _ERR_CODE	ADR esMD Letters File attachment is missing. One or more attachments is required
4	eMDR Process metadata parsing error	EMDR_PROCESS_METADATA_PARSING_ERR_CODE	Error Parsing the eMDR Process Metadata XML file.
5	File size is 0 MB or greater than 140 MB in size for Post-Pay packages	ADR_LETTER_PACKAGE_FILE_EQUALS_MIN_LIMIT_OR_EXCEEDS_MAX_LIMIT	ADR Letter Package file size is 0 MB or exceeds 140 MB
6	Attachments other than PDF format for Post-Pay and Post-Pay-Other packages	INVALID_FILE_EXTENSION_FOR_ADR_LETTER_IN_PDF	Invalid File Extension for ADR PDF Letter

ID	Scenario	Error Code	Error Message
7	Missing eMDR Structured file or ADR letter file	EMDR_STRUCTURED_FILE_OR_ADR_LETTER_IN_PDF_MISSING_OR_MULTIPLE_FOR_ADR_LETTER_PACKAGE	eMDR Structured XML File and/or ADR PDF Letter missing or more than one for ADR Letter Package
8	Structured XML file cannot be parsed	EMDR_STRUCTURED_FILE_PARSING_FAILURE	eMDR Structured XML File cannot be parsed
9	Corrupted PDF file	CORRUPTED_FILE_RCVD_FROM_HIH	PDF file with Document ID <<DOCUMENT_ID>> found to be corrupted. Correct and resubmit

11.2.15 DCF Pickup Notification

RC Client after downloading the Document Code file, generates the successful pickup notification in real time when all the edits are validated successfully. Figure 53: DCF Pickup Notification shows the sample DCF Pickup Notification.

Figure 53: DCF Pickup Notification


```
{
    "notification": [
        {
            "pickupTime": "2023-03-31T14:26:04.1610000-04:00",
            "filename":
"ES9999.V.L17.EVME0004332201EC.ESMD01.D021023.T1100100",
            "esMDTransactionId": "VME0004332201EC",
            "submissionTime": "2023-03-31T14:26:04.1620000-04:00",
            "status": "SUCCESS"
        },
            "errorMessages": [

            ]
    ],
    "senderRoutingId": "ES9999",
    "notificationType": "PICKUP"
}
```


11.2.16 DCF Error Pickup Notification

If there are any validation edits failure, RC Client API will generate error pickup notification back to esMD in real time using Notification API. Figure 54: DCF Error Pickup Notification shows the sample error pickup notification.

Figure 54: DCF Error Pickup Notification


```
{
    "notification": [
        {
            "errorMessages": [
                {
                    "errorName": "ERROR DOCUMENT CODES VALIDATE FILE",
                    "errorDescription": "Invalid line length for line 128
Expected: 1035, Actual:  1036",
```

```
                    "errorCode": "515"
                }
            ],
            "pickupTime": "2023-03-31T14:26:04.1610000-04:00",
            "filename":
"ES9999.V.L17.EVME0004332201EC.ESMD01.D021023.T1100100",
            "esMDTransactionId": "VME0004332201EC",
            "submissionTime": "2023-03-31T14:26:04.1620000-04:00",
            "status": "FAILED"
        }
    ],
    "senderRoutingId": "ES9999",
    "notificationType": "PICKUP"
}
```


11.2.17 Letters Request

The Letters request is a JSON message created in RC Client and uploaded to esMD. The response notification (success or failure) is received in real time after delivering the Letters request to HIH. esMD processes the request in deferred approach if the document (PDF attachment) size is greater than 10 MB. The statuses for these requests are retrieved using Status API. The pdf document is encoded in MD5 hexadecimal format.

The description PADL/RRL is changed to LETTERS in all the places wherever applicable.

The Qualifier Type and ID elements in the details section in the Letters Json (Figure 55: Sample Letters JSON Message) are made mandatory.

- esMD will accept and validate the common qualifier values in the LETTERS requests sent by the RCs.
- RCs must only send the Qualifier Type values which are provided in the following pre-defined list in the LETTERS request (CTC 20).
    - UTN
    - BENE-NAME
    - BENE-ID
    - BENE-DOB
    - PDOS
    - PDOS-RANGE
    - DECISION-DATE
    - UTN-EXPIRATION-DATE
    - CLAIM-ID
    - CASE-ID
    - DOS
    - DOS-RANGE
- At least one Qualifier Type and ID must be present in letters with the values.

- RC's can send multiple Qualifier Type/ID pairs in the LETTERS. The same Qualifier Type and ID pair can be repeated.
- The value for "Qualifier Type" must exactly match the valid qualifier values list and cannot contain null, empty or spaces.
- The value for "id" can be of any format and length and cannot be null, blank or contain spaces.
- A validation error message is returned for any invalid Qualifier Type or ID values.

The sample JSON message is included in Figure 55: Sample Letters JSON Message.

Figure 55: Sample Letters JSON Message


```
{
      "summaryinformation": {
            "reimbursementdenied": "2",
            "reimbursementreviewed": "wewwe",
            "totalclaimsdeniedorreduced": "sdfdsf-23324",
            "claimsreviewed": "sdfsdf 003_,",
            "totalclaimspaid": "23432er -000,",
            "errorrate": "sfsdf99----$$$",
            "claimserrorrate": "sfsdf 4343430--  "
      },
      "paprogramnameproviderexempted": [
            "Test Provider one two three"
      ],
      "enclosures": "test",
      "letterdate": "10/10/2023",
      "documentinfo": [
            {
                  "encoded_document": "YKMTQ2CiUlRU9GCg==",
                  "mimetype": "application/pdf",
                  "encoded_document_checksum":
"42fb391e4be0a3762cac1ec461777c34",
                  "documentid": "RRL-1"
            }
      ],
      "inputfield": [],
      "providersystemidentifier": [
            {
                  "senderidentifier": "test"
            },
            {}
      ],
      "letterid": "AWSAUTO5TSDHFJDSJDHFJDHS_JDF305961_000",
      "provider_or_receiverinformation": {
            "zipcode": "23234-23323",
            "telephonewithextension": "2423423223",
            "lastnameororganizationname": "SSM St Anthony's Hospital",
            "firstname": "SSM St Anthony's Hospital",
            "address2": "test",
            "city": "test",
            "ptan": "898989999",
```

```
            "address1": "fsdfdsfdsfdsfdsfdsfdsfdsfdsf",
            "npi": "1992853964",
            "letteraddressedto": [],
            "state": "ma",
            "fax": "234324"
      },
      "details": [
            {
                  "dateofbirth": "12/11/1990",
                  "beneficiaryfirstname": "sfsfd",
                  "reviewinformation": [
                        {
                              "servicetodate": "12/30/20+23",
                              "denialcode": "test3",
                              "decision": "test2",
                              "reviewfactor": "test",
                              "datainputfield": [],
                              "servicefromdate": "12/12/2023",
                              "revisedfactor": "test1",
                              "decisionrationale": "test33"
                        }
                  ],
                  "inputfield": [],
                  "qualifier": [
                        {
                              "type": "PDOS,
                              "id": "11990-20-30-$123&*23432"
                        },
                        {
                              "type": "PDOS-RANGE",
                              "id": "11990.22220-30WEWQ-$123&*23432"
                        },
                        {
                              "type": "CASE-ID",
                              "id": "CSasdasdaas234356 sfas_123"
                        },
                        {
                              "type": "BENE-NAME",
                              "id": "CSasdasdaas234356 sfas_123"
                        },
                        {
                              "type": "BENE-ID",
                              "id": "CSasdasdaas234356 sfas_123"
                        },
                        {
                              "type": "BENE-DOB",
                              "id": "12/11/1981-$%CSasdasdaas234356"
                        },
                        {
                              "type": "DECISION-DATE",
                              "id": "1190-20-30-$123&*23432"
                        },
                        {
                              "type": "UTN-EXPIRATION-DATE",
                              "id": "1190-20-30-$123&*23432"
                        },
```

```
                        {
                              "type": "CLAIM-ID",
                              "id": "CSasdasdaas234356 sfas_1231190-20-30"
                        },
                        {
                              "type": "DOS",
                              "id": "CSasdasdaas234356 sfas_1231190-20-30"
                        },
                        {
                              "type": "DOS-RANGE",
                              "id": "1190-20-30-$123&*
                        },
                        {
                              "type": "DECISION-DATE",
                              "id": "1190-20-30-$123&*23432"
                        }
                  ],                "beneficiarylastname": "sfdsf",
                  "billingperiod": [
                        {
                              "billingperiod_startdate": "01/01/2023",
                              "billingperiod_enddate": "12/05/2023",
                              "utn": "23432"
                        }
                  ],
                  "medicalrecordnumberorpatientaccountnumber": "234",
                  "beneficiaryid": "12321"
            }
      ],
      "category": "20.1",
      "subcategory": "NA",
      "senderinformation": {
            "zipcode": "12311-2344",
            "division": {
                  "type": "ssf",
                  "value": "MACsdfdsf"
            },
            "telephonewithextension": "1234567890",
            "address2": "test",
            "city": "columbia",
            "address1": "abc",
            "rctype": "sfsd",
            "state": "MD",
            "fax": "1232333331",
            "linesofbusiness": "LOBettr",
            "reviewcontractorname": "sdf"
      }
}
```

The sample validation error messages for invalid Qualifier Type and ID scenarios are shown in Table 11: Sample Validation Errors from esMD below.

Table 11: Sample Validation Errors from esMD

S.No.	Scenario	Error Description
1	Invalid Qualifier Type	{ "esmdtransactionid": "DXU0007209741EC", "routingId": "ESD002", "hihOid": "urn:oid:123.456.657.126", "rcOid": "urn:oid:2.16.840.1.113883.13.34.110.1.999.2", "rcName": "Test Review Contractor 2", "letterId": "AWSAUTO5TSDHFJDSJDHFJDHS_JDF305961_000", "contentType": "20", "status": "FAILED", "statusDescription": "esMD validation error. Please correct and resubmit.", "errorDetails": [{ "errorCode": "LETTERS_SCHEMA_000", "errorName": "", "errorDescription": "esMD validation error:/details/0/qualifier/0/type instance value (\"CASE-I\") not found in enum (possible values: [\"UTN\",\"BENE-NAME\",\"BENE-ID\",\"BENE-DOB\",\"PDOS\",\"PDOS-RANGE\",\"DECISION-DATE\",\"UTN-EXPIRATION-DATE\",\"CLAIM-ID\",\"CASE-ID\",\"DOS\",\"DOS-RANGE\"])"
2	Invalid Qualifier ID	"errorDescription": "esMD validation error:/details/0/qualifier/0/id instance type (null) does not match any allowed primitive type (allowed: [\"string\"])"
3	Missing Qualifier Id element	"errorDescription": "esMD validation error:/details/0/qualifier/0 object has missing required properties ([\"id\"])"
4	Qualifier Id has all spaces entered in the letters	"errorDescription": "The qualifier id does not accept spaces. Correct and resubmit."

12. RC Client Components

Figure 56: RC Client Components shows the internal components of RC Client application. The following sections describe each component in detail.

Figure 56: RC Client Components

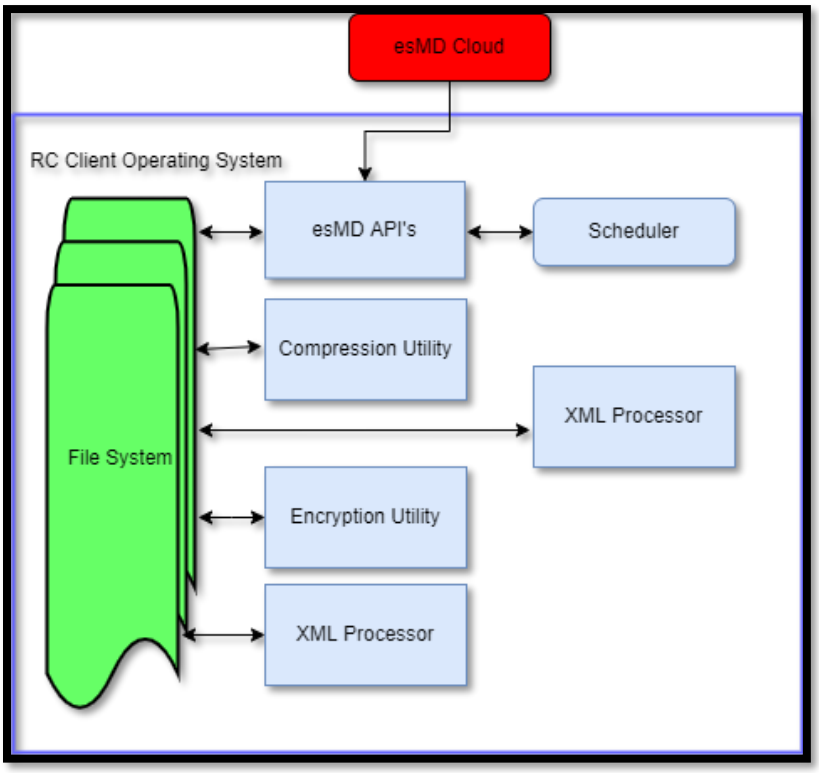
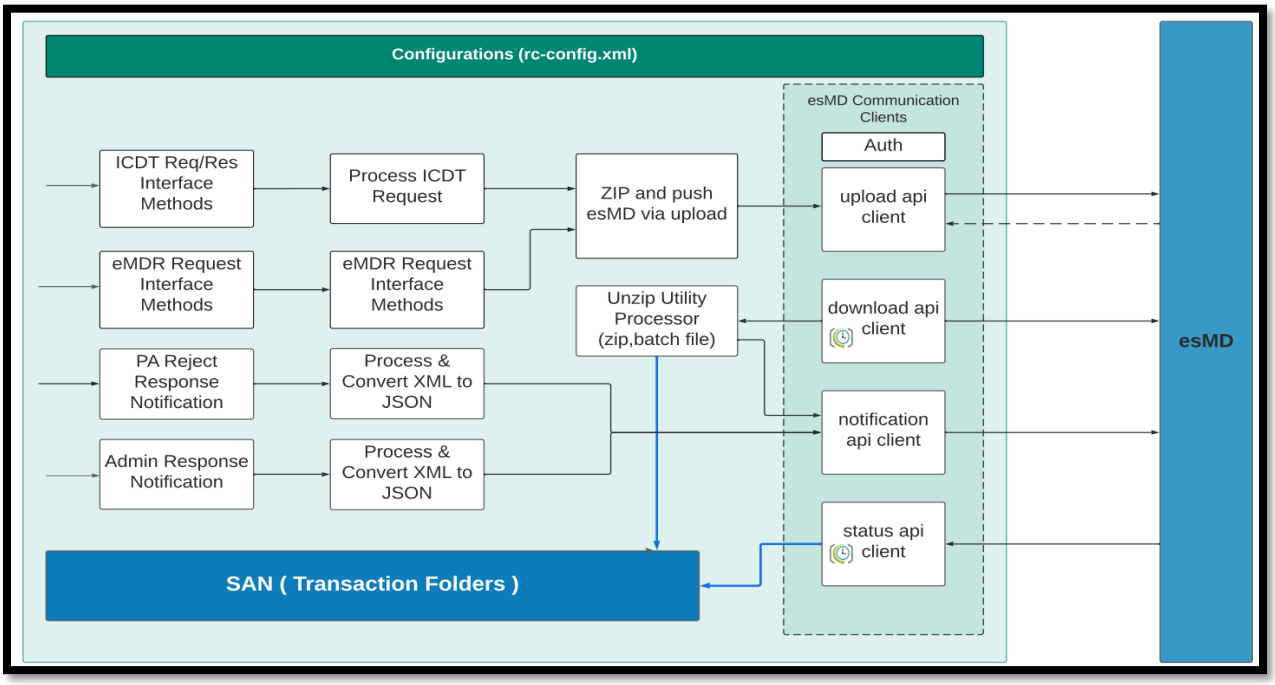


Figure 57: RC Client API illustrates the API's which are implemented in the esMD Cloud.

Figure 57: RC Client API



12.1 esMD APIs

The following are the esMD APIs which are used to authenticate the user, transfer files between the esMD and RC Client, send notifications to the HIH in real time, and send acknowledgments/responses from the HIH.

1. Auth API
2. Upload API
3. Download API
4. Notification API
5. Status API
6. Upload Realtime API

12.2 Compression Utility

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

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

12.3 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 24.1 Security.

12.4 XML Processor

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

12.5 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 in regular intervals determined by the “checkFrequency” parameter in the XML Configuration File.

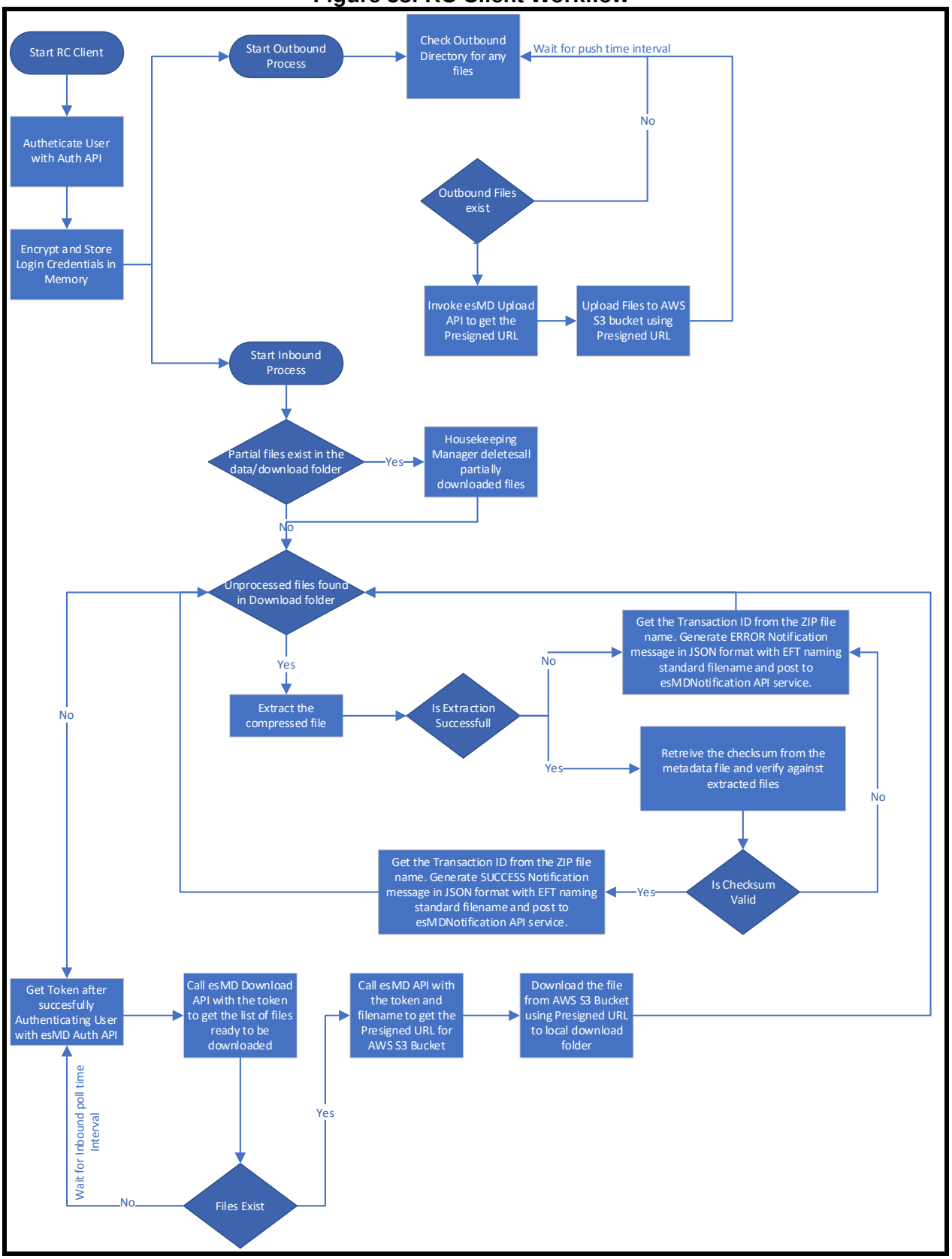
12.6 Housekeeping Manager

The Housekeeping Manager allows the RC Client to recover from any abnormal terminations. In this situation, the RC Client does not enable recovery, and the RC must contact the esMD Service Desk.

13. RC Client Workflow

The workflow associated with Figure 56: RC Client Components is broken down in Figure 58: RC Client Workflow, followed by a detailed description of the workflow.

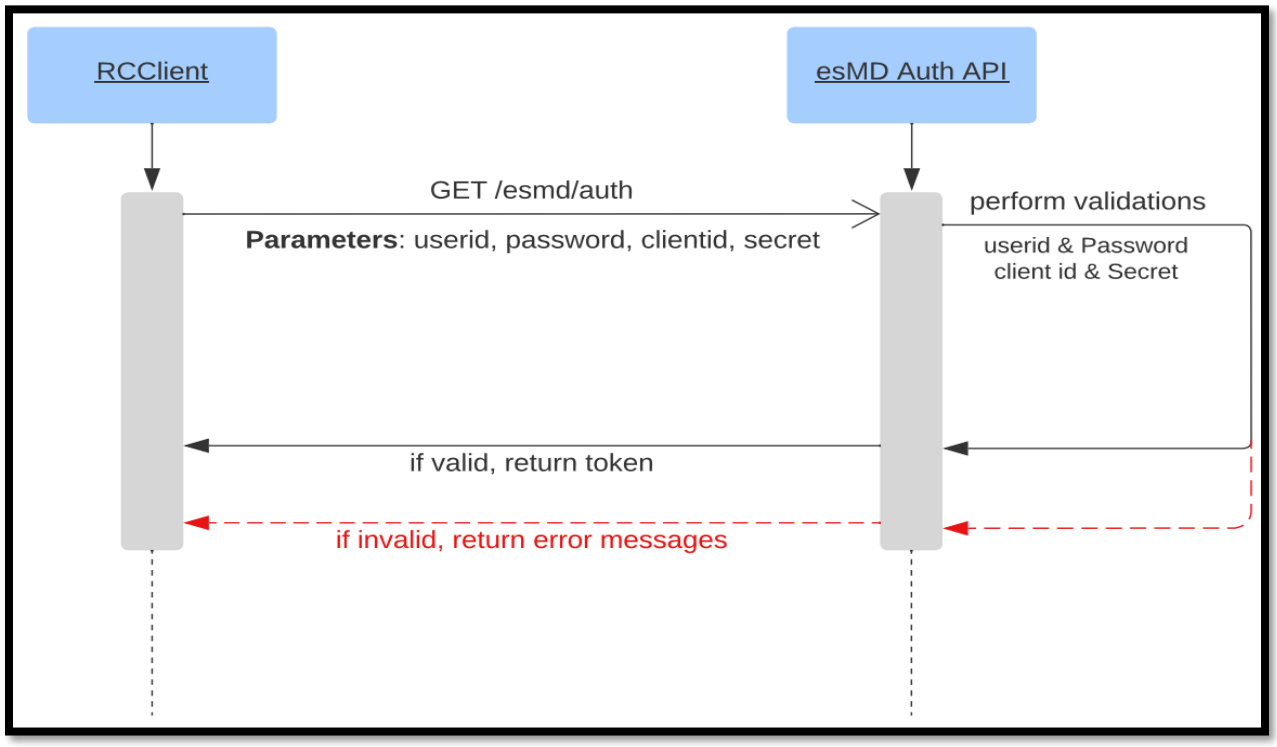
Figure 58: RC Client Workflow



14. Auth API

1. The esMD Authentication API is used to authenticate the RC Client user by verifying the username, password, mailbox, scope, client id, and client secret. The Auth API receives a token after the user is successfully authenticated. The token is used by other APIs to upload and download files from esMD Cloud.
2. If the user is not found in the system, the API returns an error message back to the user.
3. The Authentication Token is valid for only 30 minutes after the user makes the API call.
4. If the Authentication Token expires, the user is allowed to make another Auth API call.

Figure 59: esMD Auth API Sequence Diagram



14.1 Auth Endpoint URL:

https://val.cpiapigateway.cms.gov/api/esmd/ext/v1/auth/generate

Method: GET

14.2 Auth API Request Body

None

14.3 Auth API Request Parameters

Table 12: Auth API Request Parameters

Name	Description	Type	Data Type	Required
uid	User ID	header	string	Yes
password	Password	header	string	Yes
clientid	Client ID to authorize the user	header	string	yes
clientsecret	Client Secret to authorize the user	header	String	Yes
scope	Scope to limit user access to specific API. EX: devesmdrc/download	Header	String	yes
mailbox_id	User specific Mailbox ID. EX: ES9996	Header	String	yes

14.4 Auth API Request

Figure 60: Auth API Request

```
GET /esmd/v1/auth
uid:testuserid
password: xxxxxx
clientid: xxxxxxxx
clientsecret: xxxxxxxx
scope: devesmdrc/download
mailbox_id: ES9996
```

14.5 Auth API Success Response

Figure 61: Auth API Success Response

```
HTTP/1.1 200 Accepted
{
  "access_token":"xxxxxxxxxxxxxxxxxx",
  "token_type":"Bearer",
  "expires_in":3600
}
```

14.6 Auth API Failure Response

Figure 62: Auth API Failure Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json
Cache-Control: no-store
{
    "error": "Authentication Failed.Invalid UserID/Password"
}
```

15. Upload API

The Upload API allows the RC Clients to upload the files to esMD Cloud. The Authentication Token is generated through Auth API, and once the authentication is successful, the review contractors can upload the files successfully using the pre-signed URL generated with filename and token.

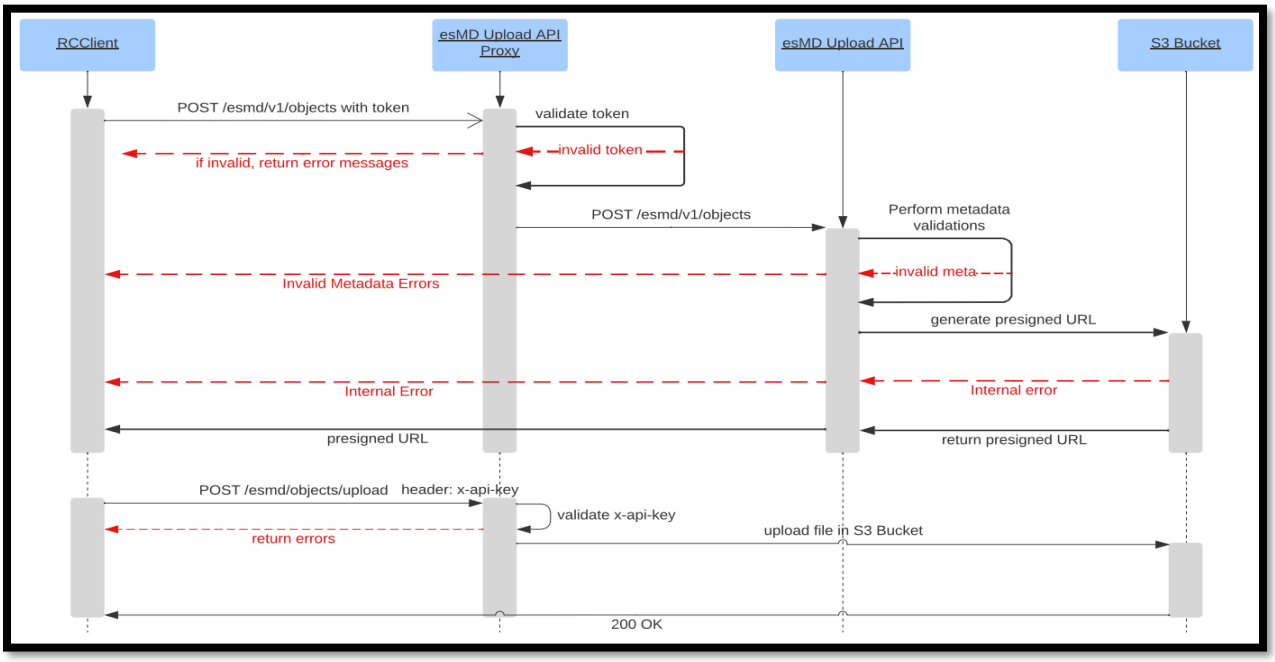
If the metadata validations fail, the Upload API rejects the request and sends an error message.

If the Upload API call is successful, a pre-signed URL is generated with the following metadata elements:

- Filename
- contentchecksum
- content type
- senderroutingid
- scope
- size
- uid
- Authorization

1. The pre-signed URL is valid for 15 minutes for the user to upload the documents.
2. The system cross-validates the metadata information against the file name with the pre-signed URL and the files are uploaded to the esMD once the validation is successful.
3. If the cross-validation between the pre-signed URL and metadata information fails, an error notification is generated and sent to the RC.

Figure 63: esMD Upload API Sequence Diagram



15.1 Upload API Endpoint URL

https://val.cpiapigateway.cms.gov/api/esmd/ext/v1/objects

Method: POST

15.2 Upload API Request Body

None

15.3 Upload API Request Parameters

Table 13: Upload API Request Parameters

Name	Description	Type	Data Type	Required
senderroutingID	Sender Routing ID	header	string	Yes
scope	Scope	header	string	Yes
content-type	Application/Fileformat	header	string	Yes
Authorization	Bearer Token	header	string	Yes

Name	Description	Type	Data Type	Required
Content-md5	Checksum is autogenerated and received in md5 binary converted to base 64 format	header	string	Yes
filename	Filename	header	string	Yes
senderroutingid	Sender Routing ID	header	string	Yes
contentchecksum	Checksum(auto generated)	header	string	Yes
size	Content Length	header	String	Yes

15.4 Upload API Request

Figure 64: Upload API Request

```
POST /esmd/v1/objects

size: 100 mb
Content-md5: xdfdsasdfsfrerqewq
Content-type: application/zip
filename: ESMD2.D.L1_5.GUID01020304051.ES0001.D040622.T1051500.zip
Authorization:
eb0c4513bb3d8b265e4f7a33cebeb61f5c527a74b8d516efe8465e2378cc5853
uid:testuserid
senderroutingid: ES0001
contentchecksum: xdfdsasdfsfrerqewq
scope: devesmdrc/upload
```

15.5 Upload API Response (esMD Received)

Figure 65: Upload API Response (esMD Received)

```
application/json
{
    "status": "SUCCESS",
    "message": "FILE UPLOADED SUCCESSFULLY",
    "filename": "esd001.D.guid1234.esmd2.D123234.T121212.zip"
}
```

15.6 Upload API Error Response (Upload Failed)

Figure 66: Upload API Error Response (Upload Failed)

```
HTTP/1.1 403 Forbidden
<?xml version="1.0" encoding="UTF-8"?>
<Error>
<Code>SignatureDoesNotMatch</Code>
<Message>The request signature we calculated does not match the signature
you provided. Check your key and signing method.</Message>
<AWSAccessKeyId>ASIAIM2D54PBBIG5CC4Q</AWSAccessKeyId>
<StringToSign>AWS4-HMAC-SHA256
20171016T211404Z
20171016/us-east-1/s3/aws4_request
b5f99d8ba1d5e4bffc4307f16c0cca6b5614e0aaca70fe56c065dbe7c59d4095</StringToSi
gn>
<SignatureProvided>5504047ce06d74ea583241fae6a5bed4fea8faaa932a33aeacd6fc7f6
334573</SignatureProvided><StringToSignBytes>41 57 53 34 2d 48 4d 41 43 2d
53 48 41 32 35 36 0a 32 30 31 37 31 30 31 36 54 32 31 31 34 30 61 63 61 37
30 66 65 35 36 63 30 36 35 64 62 65 37 63 35 39 64 34 30 39
35</StringToSignBytes>
```

15.7 Uploaded File Format

<<ReceiverRoutingId>>.
<<environment>>.L<<CTC>>.<<DeliveryType>>GUID.<<SenderRoutingID>>.DMMddy.THHmmssS.zip

15.8 Elements Description Inside the File Name

- Environment: Dev – D, Val – V, UAT – T, PROD – P
- Receiver Routing ID: RC/esMD Mailbox/sub_id (Ex: ESMD2)
- CTC: Content Type Code (associated CTC code, numeric and period allowed characters)
- DeliveryType: transaction types (is 1char and allowed value is Alphabet only) Ex: (Q- ICDT Request. R-> ICDT Solicited response/unsolicited response)
- GUID: esMD Transaction ID or Global Unique Id (15 AN RC Client generated random id)
- Sender Routing ID: RC/esMD Mailbox
- DMMddyy: current date in Mmddyy format
- THHmmssS: current time in HHmmssS format

File Name Example:
ESMD2.L11_2.EGUID01020304051.ESMD01.D040322.T0732220.zip

Note : Please refer to Section 6.2.1.5 of the esMD Rest Approach ICD document for more examples of the request and response messages and JSON structures.

16. Download API

The Download API is used to download the files from esMD cloud. The API gets the list of files ready to be downloaded using the token returned by the Auth API. The Download API gets a pre-signed URL for each file and downloads the file using the pre-signed URL.

1. The user can send the request to the Download API to get the list of files available for download in esMD with the valid Metadata provided below in the API request:
   - uid
   - senderroutingid
   - authorization

2. Once the Download API call is successful, the list of files available for download will be sent to the user.
3. The Download API will reject the API request and sends the error message if the Metadata validations are failed.
4. If the file name is validated successfully, Pre-signed URL is generated for the filename.
5. The Filename should be sent in the below format in the Download API request:

File Format:

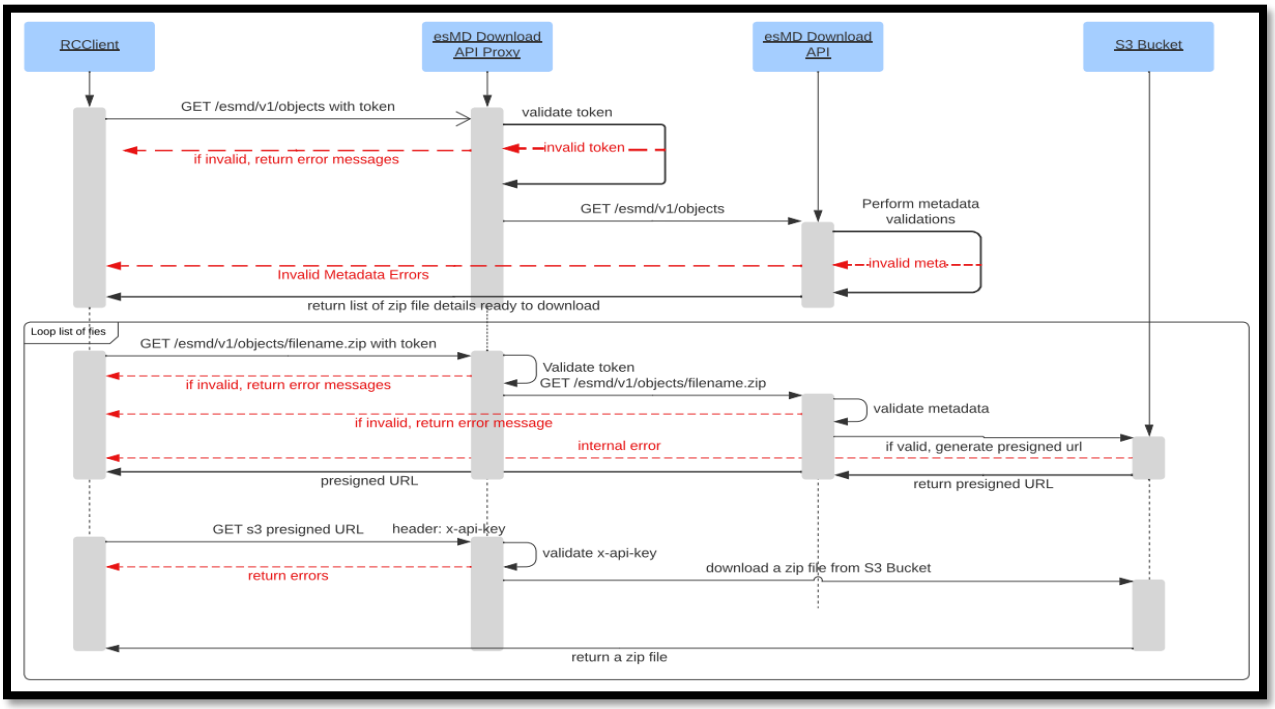
<<ReceiverRoutingId>>.<<environment>>.L<<CTC>>.<<Delivery_type>>GUID.<<SenderRoutingID>>.DMMddyy.THHmmssS.zip

- Elements Description Inside the File Name:
- Environment: Dev – D, Val – V, UAT – T, PROD – P
- Receiver Routing ID: RC/esMD Mailbox/sub_id (Ex: ESD0001)
- CTC: Content Type Code (associated CTC code, numeric and period allowed characters)
- DeliveryType: transaction types (E – xdr/x12 package, Q – ICDT Request …etc) (is 1char and allowed value is Alphabet only)
- GUID: esMD Transaction ID or Global Unique Id (15 AN RC Client generated random id)
- Sender Routing ID: RC/esMD Mailbox
- DMMddyy: current date in Mmddyy format
- THHmmssS: current time in HHmmssS format

File Name Example:
ES0001.D.L11_2.EGUID01020304051.ESMD01.D040322.T0732220.zip

6. The RC will receive an attachment in .zip format and must be able to download the files.
7. The pre-signed URL is valid for 15 minutes for the user to download the documents.

Figure 67: esMD Download API Sequence Diagram



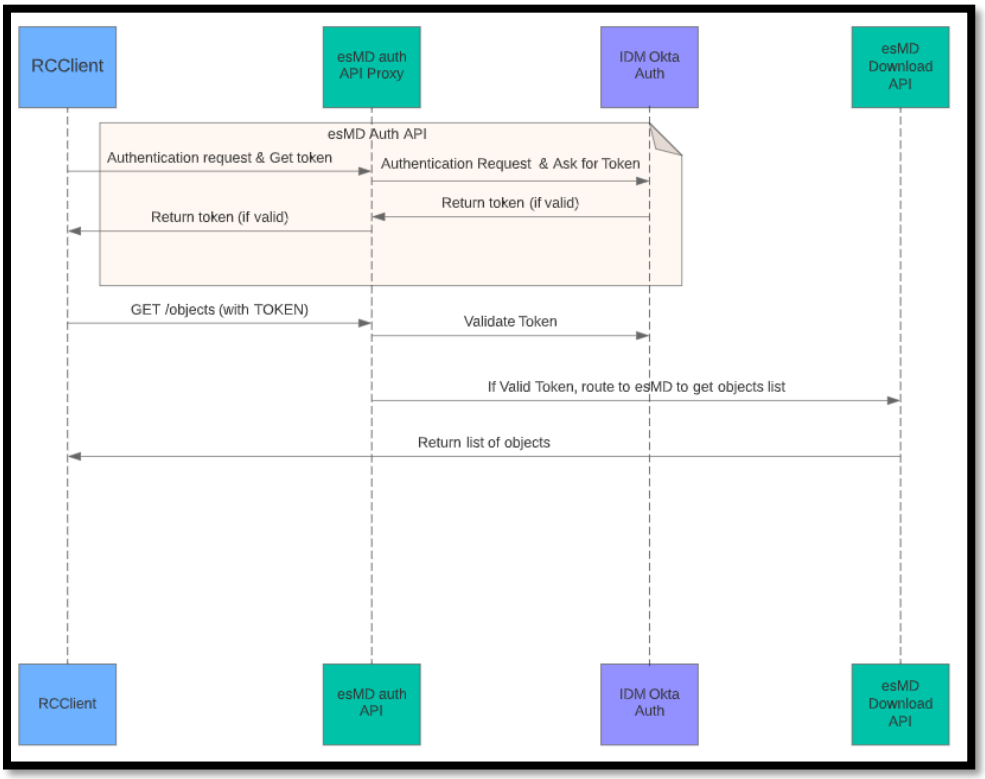
16.1 Download API Endpoint URL (Get List of Files)

https://val.cpiapigateway.cms.gov/api/esmd/ext/v1/objects

Method: GET

This API is used to get the list of files ready to download in esMD.

Figure 68: esMD GetListOfFiles Sequence Diagram



16.2 Download API Request Body (Get List of Files)

None

16.3 Download API Request Parameters (Get List of Files)

Table 14: Download API Request Parameters (Get List of Files)

Name	Description	Type	Data Type	Required
uid	User ID	header	string	Yes
senderroutingID	Sender Routing ID	header	string	yes
authorization	Sender Organization ID	header	string	Yes
scope	Scope	header	string	Yes

16.4 Download API Response

Figure 69: Download API Response

```
application/json
200 OK
Success
```

16.5 Download API Example

Figure 70: Download API Example

```
Request:
GET /esmd/v1/objects

uid: {uid}
senderroutingid: ESD001
authorization: xxxxxxxxxxxxxxxxxx
scope: devesmdrc/download
```

16.6 Download API Success Response

Figure 71: Download API Success Response

```
HTTP/1.1 200 OK
{
    "status": "SUCCESS",
    "message": "",
    "objects": [
        {
            "filename": "T#EFT.ON.D.SD002.L7.EAZV.D070122.T0640010",
            "size": "",
            "statusFlag": "",
            "createdOn": "2022-07-01T06:39:49.279534",
            "id": "",
            "lastDownloaded": ""
        }],
    "count": 1
}
```

16.7 Download API Error Response

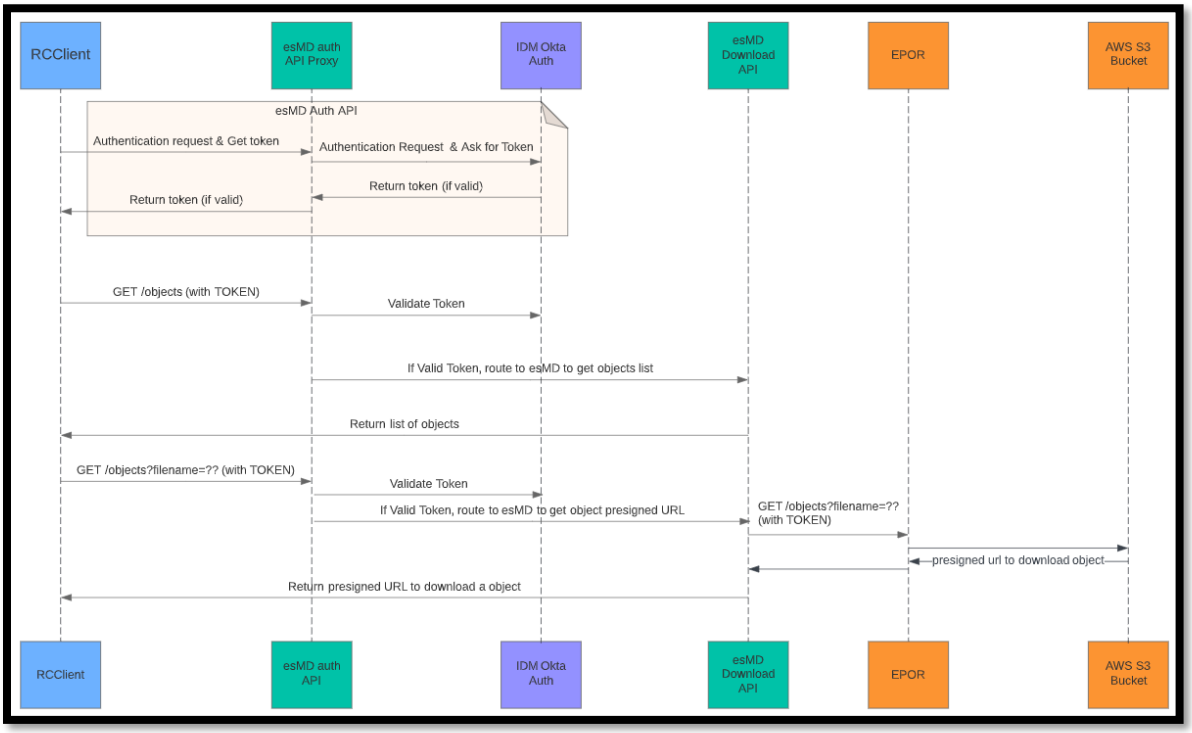
Figure 72: Download API Error Response

```
HTTP/1.1 404 Not Found
{
    "status": "SUCCESS",
    "message": "NO FILES ARE AVAILABLE TO DOWNLOAD FROM ESMD AT THIS TIME”
    "objects": [],
    "count": 0
}
```

16.8 Download API Generate Presigned URL for ZIP File

This API returns the pre-signed URL to download a zip file from esMD Cloud.

Figure 73: esMD Generate Presigned URL Sequence Diagram



16.9 Download API Parameters (Download File)

Table 15: Download API Parameters (Download File)

Name	Description	Type	Data Type	Required
uid	User ID	header	string	Yes
senderroutingID	Sender Routing ID	header	string	yes
authorization	Sender Organization ID	header	string	Yes
scope	Allowed values for environment are: uat or prod (case sensitive)	header	string	Yes

16.10 Download API (Download File) Endpoint URL

https://val.cpiapigateway.cms.gov/api/esmd/ext/v1/objects/ESD002.D.L1.AAQ000193750 1EC.ESMD2.D061622.T0423540.zip

Method: GET

16.11 Download API Response (Review Contractor Received)

Figure 74: Download API Response (Review Contractor Received)

```
application/json

uid: {uid}
senderroutingid: ESD001
authorization: xxxxxxxxxxxxxxxxxx
scope: devesmdrc/download
```

16.12 Download API Response (Download File Success)

Figure 75: Download API Response (Download File Success)

```
HTTP/1.1 202 Accepted
{
{

    "status": "SUCCESS",
    "message": " PRESIGNED URL IS GENERATED",
    "contents": [
        {
                "filename":
"ESD002.D.L1.AAQ0001937501EC.ESMD2.D061622.T0423540.zip",
"url": "https://esmdcloud-
dev.cms.hhs.gov:8089/objects/upload/ESMD2.D.L1_6.UYEO11323312456.ES9996.D012
711.T1523363.zip?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Date=20220705T182842Z&X-Amz-SignedHeaders=content-md5%3Bcontent-
type%3Bhost&X-Amz-Expires=899&X-Amz-
Credential=AKIAV5F4F2A2VLATR5P6%2F20220705%2Fus-east-
1%2Fs3%2Faws4_request&X-Amz-
Signature=29fecbef0d0eef881529dbc666ad99da76f98b32273c6f55bcf5481d7217c88b",
                "lastDownloaded": "",
                "comments": ""
        }
    ]
}
```

16.13 Download API Error Response

Figure 76: Download API Error Response

```
HTTP/1.1 404 Not Found

{
{

    "status": "FAILED",
    "message": "VALIDATION FAILED",
```

```
    "filename": "",
    "errorDetails": [{
        "errorCode": "E0001",
        "errorMessage": "Required header missing"
    }]
}
```


Note : Please refer to Section 6.3.1.5 of the esMD Rest Approach ICD document for more examples of the request and response messages and JSON structures.

17. Notification API

When the RC downloads a package for processing, the Notification API is invoked to send either a Successful or an Error pickup notification to the HIH. RCs receive the Acknowledgement for the Pickup Notification in real time.

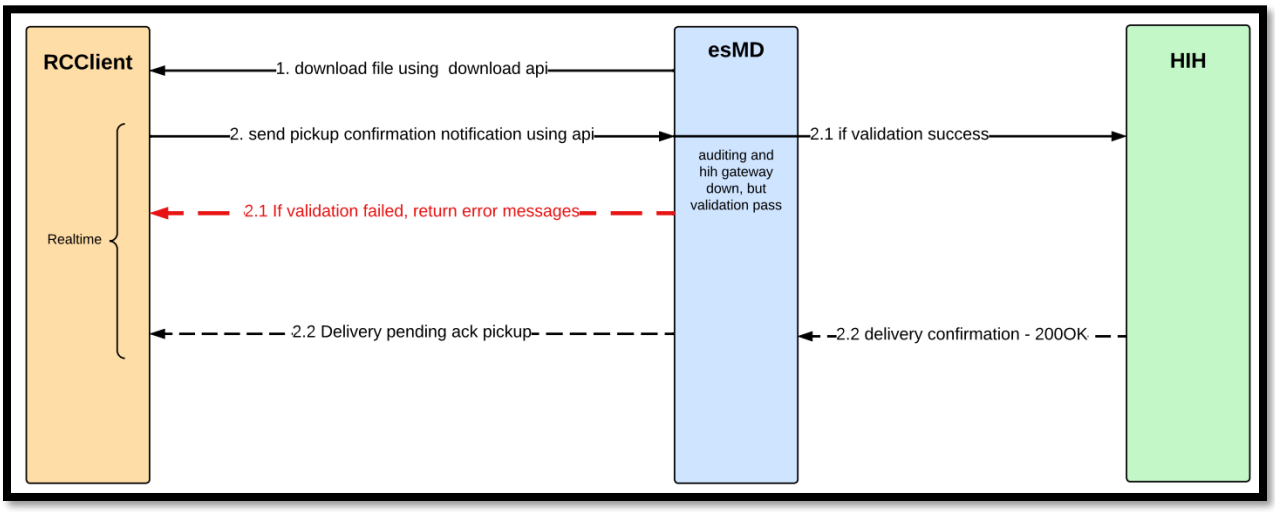
The Notification API is used to process the messages below from the RC to the HIH:

- Pickup Success Notification
- Pickup Failure Notification
- PA reject response
- Admin Error
- ICDT Pickup Notification
- ICDT Admin Error

The PA Reject Response and Admin Error notifications are processed asynchronously using a scheduler which runs at specific time intervals. Currently, the notification API runs when the inbound or outbound process is enabled and uses the “pushFrequency” value from the RC client configuration file to send the notifications.

There are no new implementation changes to the RC Client application corresponding to the esMD CAQH REST API implementation. The only change is the status description value in the Pickup, Admin and PA Reject Success responses received from esMD for the X12 transactions. Refer the Pickup Success Response JSON message below for the updated message.

Figure 77: esMD Notification API Sequence Diagram



17.1 Notification API Endpoint URL

https://val.cpiapigateway.cms.gov/api/esmd/ext/v1/objects/notification

Method: POST

17.2 Notification API Request Parameters

Table 16: Notification API Request Parameters

Name	Description	Type	Data Type	Required
Authorization	Token	header	string	Yes
scope	Scope	header	string	Yes
content-type	Content Type	header	string	Yes

17.3 Notification API Request

Figure 78: Notification API Request

```
POST /esmd/v1/objects

Content-type: application/json
filename: ESMD2.D.L1_5.GUID01020304051.ES0001.D040622.T1051500.zip
Authorization:
eb0c4513bb3d8b265e4f7a33cebeb61f5c527a74b8d516efe8465e2378cc5853
scope: devesmdrc/notification
```

17.4 Notification API Pickup Notification from RC to HIH

Figure 79: Notification API Pickup Notification from RC to HIH

```
{
  "notificationType": "PICKUP",
  "senderRoutingId": "ESD002",
  "contenttypecd" : "1",
  "notification": [
    {
      "esMDTransactionId": "KSZ0002296501EC",
      "pickupTime": "2022-07-14T14:46:32.9061123-04:00",
      "submissionTime": "2022-07-14T14:46:32.9031133-04:00",
      "filename": "ESD002.D.L1.KSZ0002296501EC.ESMD2.D072022.T2152490.zip",
      "status": "Success",
      "errorMessages": [

      ]
}
```

17.5 Notification API Pickup Success Response

Figure 80: Notification API Pickup Success Response

```
application/json

HTTP/1.1 200 Ok
```

```
{
    "senderRoutingId": "ESD002",
    "statusDetails": [
        {
            "esMDTransactionId": "WEB0002251201EC",
            "contenttypecd" : "1",
            "status": "Success",
            "statusDescription": " esMD processed the [notification type]
successfully. The HIH Delivery notification will be sent after the HIH
delivery confirmation is received.",
            "errorMessages": []
        }
    ]
}
```


Note : The response message is same for **Pickup** and **Admin.** The "[notification type]" values are PA REJECT, ADMIN and Pickup.

17.6 Notification API Pickup Failure Response

Figure 81: Notification API Pickup Failure Response


```
HTTP/1.1 202

{
    "senderRoutingId": "ESD002",
    "statusDetails": [
        {
            "esMDTransactionId": "KSZ0002296501EC",
            "contenttypecd": "",
            "status": "Failed",
            "statusDescription": "Metadata Validation Failed",
            "errorMessages": [
                {
                    "errorCode": "ESMD_PROCESSED_PICKUP_NOTIFICATION",
                    "errorName": "",
                    "errorDescription": "ESMD validation error: esMD
processed pickup notification for this transaction"
                }
            ]
        }
    ]
}
```


Note : Please refer to Section 6.4.1.4 of the esMD Rest Approach ICD document for more examples of the request and response messages and JSON structures.

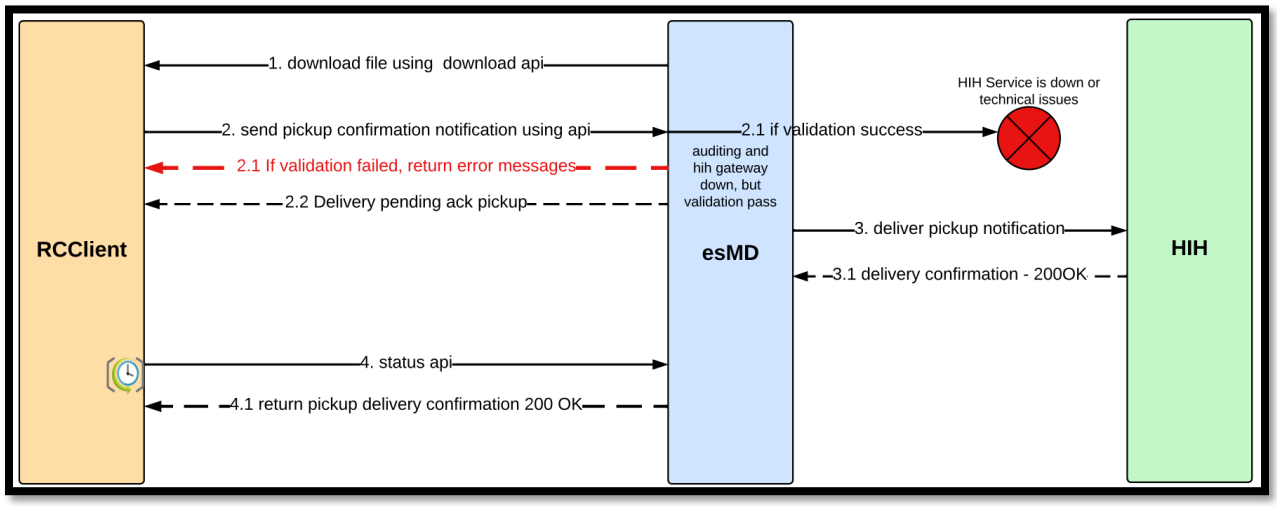
18. Status API

The Status API is a scheduled process that runs at specific time intervals and retrieves the status from esMD for the scenarios below. It also updates the RC on the successful notification delivery status to the HIH/RC:

- Pickup Success Notification
- Pickup Failure Notification
- PA reject response
- Admin Error
- HIH Delivery Notification
- ICDT
  a. Validation Errors
  b. Acknowledgement

The Status messages are processed asynchronously using a scheduler which runs at specific time intervals. Currently, the Status API runs when the inbound or outbound process is enabled and uses the “pushFrequency” value from the configuration to receive statuses.

Figure 82: esMD Status API Sequence Diagram



18.1 Status API Endpoint URL

https://val.cpiapigateway.cms.gov/api/esmd/ext/v1/objects/status

Method: GET

18.2 Status API Request Body

None

18.3 Status API Request Parameters

Table 17: Status API Request Parameters

Name	Description	Type	Data Type	Required
Authorization	Token	header	string	Yes
content-type	Content Type	header	string	Yes
senderroutingid	Sender RoutingID	header	string	Yes
uid	Username	header	string	Yes

18.4 Status API Request

Figure 83: Status API Request

```
POST / api/esmd/v1/objects/status/rc

Content-type: application/json
filename: ESMD2.D.L1_5.GUID01020304051.ES0001.D040622.T1051500.zip
Authorization: eb0c4513bb3d8b265e4f7a33cebeb61f5c527a74b8d516efe8465e2378cc5853
scope: devesmdrc/status
```

18.5 Status API Success Response

Figure 84: Status API Success Response

```
application/json

HTTP/1.1 200
{
    "status": "SUCCESS",
    "message": "THE FOLLOWING objects ARE delivered to HIH at this point of
time",
    "statusDetails": [
        {
    "notification_type": "PICKUP",
    "delivery_type": "N",
    "esmdtransactionid": "ETI0005027701EC",
    "status": "Success",
    "statusDescription": "SENT PICKUP STATUS TO HIH"
      },
      {
    "notification_type": "ADMIN",
    "delivery_type": "N",
    "esmdtransactionid": "ETI0005027701EC",
    "status": "Success",
    "statusDescription": "SENT PICKUP STATUS TO HIH"
      },
      {
    "notification_type": "PAREJECT",
    "delivery_type": "N",
```

```
    "esmdtransactionid": "ETI0005027701EC",
    "contenttypecd": 1
    "statusDescription": "SENT PICKUP STATUS TO HIH"
      }
      ],
    "count"" 3
}
```


18.6 Status API Error Response

Figure 85: Status API Error Response


```
{
    "message": "client sent an invalid request, such as lacking required
request header or parameter"
}
```


Note : Please refer to Section 6.5.1.3 of the esMD Rest Approach ICD document for more examples of the request and response messages and JSON structures.

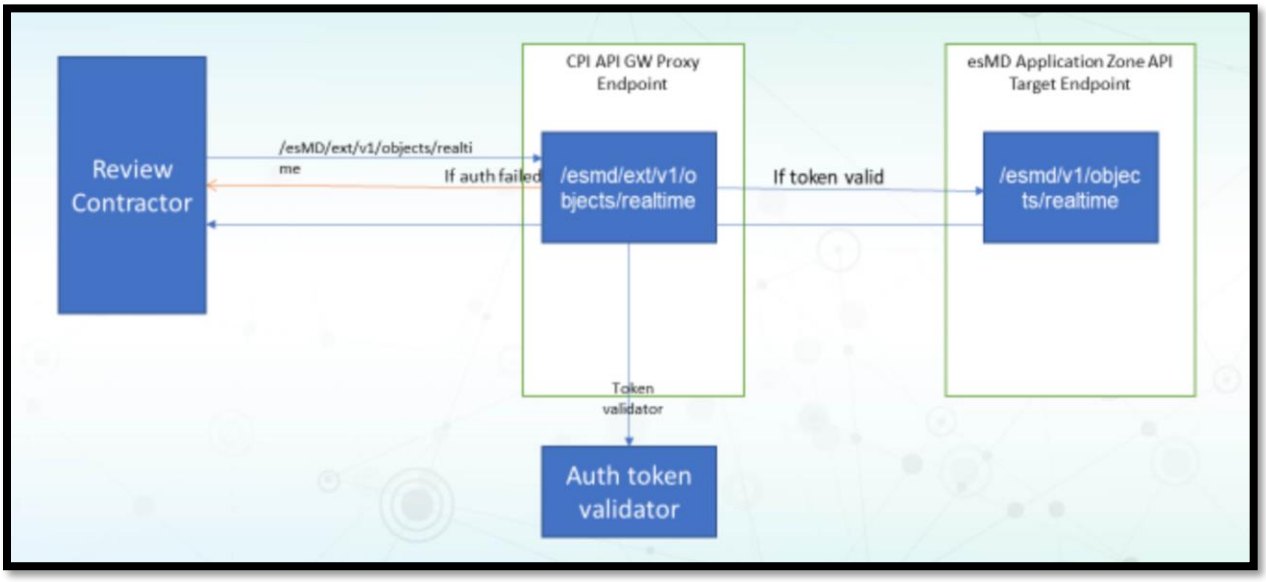
19. Upload Realtime API

The Upload Realtime API allows the RC Clients to upload the Letters files to esMD Cloud in real time. The Authentication Token is generated through Auth API, and once the authentication is successful, the Review Contractors can upload the Letters JSON messages successfully using the token. The response (HIH Delivery notification or Validation Errors) from esMD is received in real time and is saved to the users data folder in JSON format.

The esMD application uses a deferred approach if the RRL JSON request message size is greater than 10 MB. esMD validates the request and sends it to virus scanning and sends either an error or successful notification back to the RC. The HIH Delivery notifications for the large requests are retrieved asynchronously using the Status API. The validation errors are saved in data/error/RRL folder and notification files are saved in data/notification/RRL folder in JSON format.

On any authentication/authorization or communication issues while uploading to esMD, the RRL JSON messages are saved in the data\output user directory. These saved files are re-processed by the outbound process at scheduled time intervals.

Figure 86: esMD Upload Realtime API Sequence Diagram



19.1 Upload Realtime API Endpoint URL

https://val.cpiapigateway.cms.gov/api/esmd/ext/v1/objects/realtime

Method: POST

19.2 Upload Realtime API Request Body

JSON message

19.3 Upload Realtime API Request Parameters

Table 18: Upload Realtime API Request Parameters

Name	Description	Type	Data Type	Required
senderroutingid	Sender Routing ID	header	string	Yes
contenttypecode	content type code	header	string	Yes
Authorization	Bearer Token	header	string	Yes
senderroutingid	Sender Routing ID	header	string	Yes
contentchecksum	Checksum(MD5 Hex)	header	string	Yes
size	Content Length	header	String	Yes
uid	User ID	Header	String	Yes

19.4 Upload Realtime API Request

Figure 87: Upload Realtime API Request

```
POST /esmd/v1/objects/realtime

size: 1 mb
Content-md5: xdfdsasdfsfrerqewq
Authorization:
eb0c4513bb3d8b265e4f7a33cebeb61f5c527a74b8d516efe8465e2378cc5853
uid:testuserid
senderroutingid: ES0001
contentchecksum: xdfdsasdfsfrerqewq
contenttypecode: 20
letterid: UI876D876KLLH
```

19.5 Upload Realtime API Response (esMD Received)

Figure 88: Upload Realtime API Response (esMD Received)

```
application/json
```

```
{
    "esmdtransactionid": "YPR0007151172EC",
    "routingId": "ESD002",
    "hihOid": "urn:oid:123.456.657.126",
    "rcOid": "urn:oid:2.16.840.1.113883.13.34.110.1.999.1",
    "rcName": "DATS",
    "letterId": "Asdwe",
    "contentType": "20",
    "status": "SUCCESS",
    "statusDescription": "PA Decision letters or Review results letter
Processed and delivered to HIH.",
    "errorDetails": []
}
```

20. ICDT Request/Response Business Process Flow

This section describes the process flow of the ICDT Request and ICDT Solicited/Unsolicited Response sent from one RC to another RC via the esMD application. Figure 89: ICDT Request/Response Business Process Flow Diagram shows the process flow of ICDT Request and Response, and Table 19: ICDT Request/Response Business Process Flow Steps provides the detailed steps.

Figure 89: ICDT Request/Response Business Process Flow Diagram

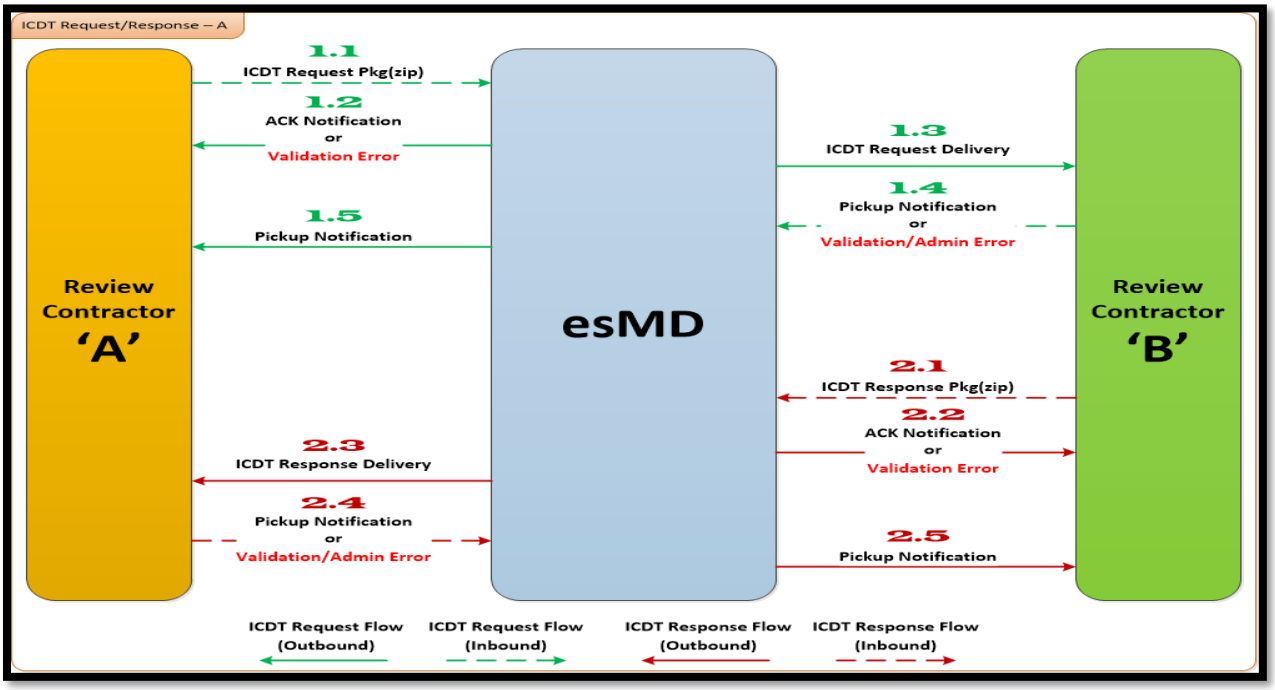


Table 19: ICDT Request/Response Business Process Flow Steps

Message Sequence	Description
1.1	Review Contractor 'A' creates the ICDT Request package, which consists of ICDT Request XML file, and sends the package to esMD via the esMD Upload API to be delivered to another RC.
1.2	esMD sends the Acknowledgement notification to the Review Contractor 'A' if the validation of the ICDT Request package is successful or the esMD system sends the validation errors for any failures. These messages are delivered to the RC by esMD Status API.
1.3	esMD delivers the ICDT Request package to the Review Contractor 'B' if the validation is successful.
1.4	Review Contractor 'B' downloads the ICDT Request Package and sends the Successful Pickup notification, Error pickup notification. Admin Errors via esMD Notification API in real time to esMD.
1.5	esMD delivers the Pickup notification, error pickup notification, or admin error to the Review Contractor 'A' using esMD Status API..

Message Sequence	Description
2.1	Review Contractor ‘B’ sends the ICDT Response package to the esMD system.
2.2	esMD validates the Response package and sends the acknowledgement back to the Review Contractor ‘B’ if the validation is successful or validation errors in case of failures. These messages are delivered to Review Contractor B using esMD Status API.
2.3	esMD system delivers the ICDT Response package to the Review Contractor ‘A’.
2.4	The RC sends the successful pickup notification, error pickup notification, or admin error to the esMD system via esMD notification API..
2.5	esMD system validates and delivers the Pickup notification, error pickup notification, or admin errors to the Review Contractor ‘B’ using esMD Status API.

21. Service Registration Processing Overview

Table 20: Service Registration Flow Steps describes the typical Service Registration flow interaction as shown in Figure 90: Service Registration Process Flow.

Figure 90: Service Registration Process Flow

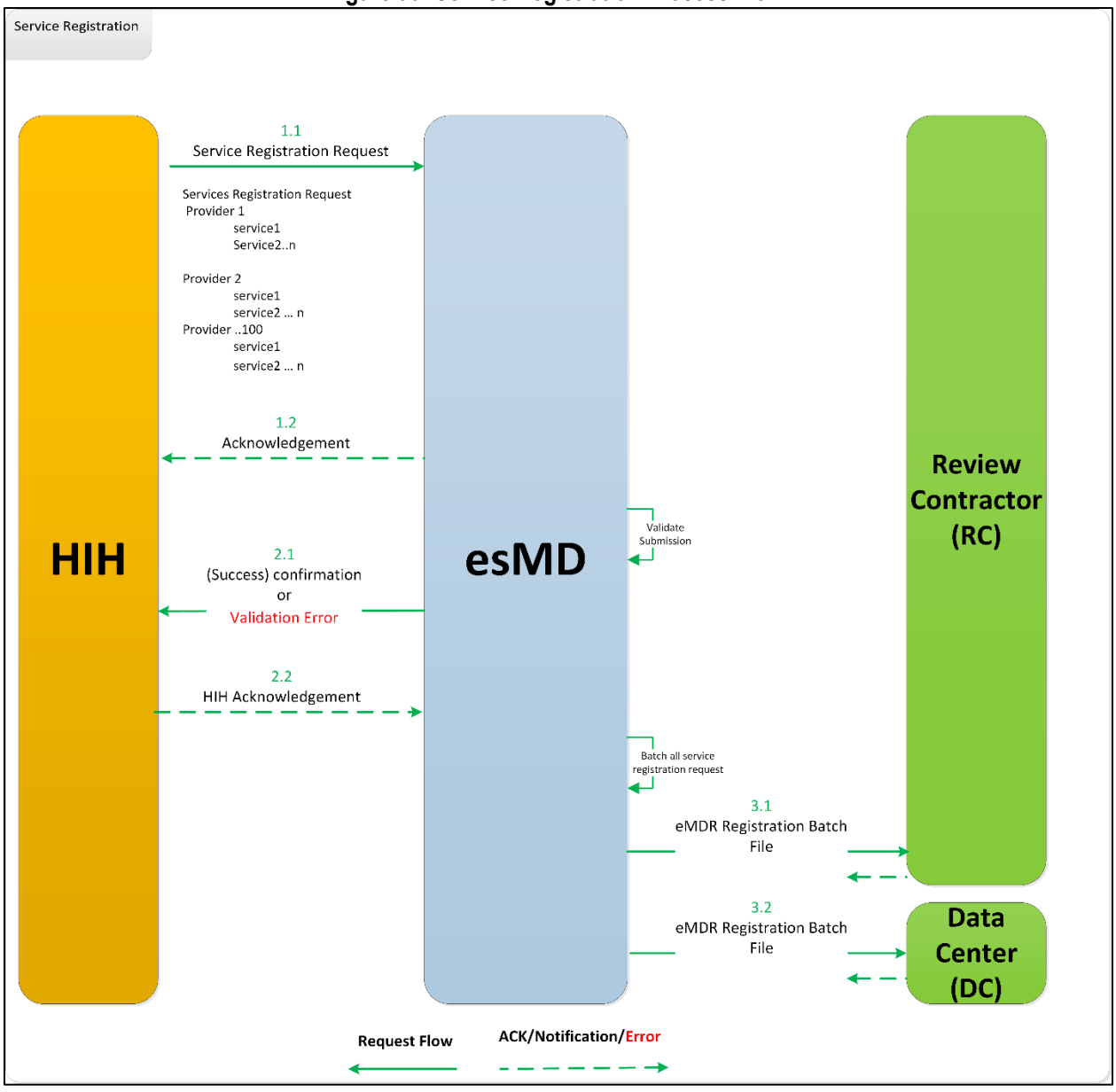


Table 20: Service Registration Flow Steps

Message Sequence	Description
1.1	The HIH submits the service registration request in XDR format to esMD with all necessary metadata information and the Service registration XML (consisting of information for one or more Provider(s) and Service(s)) wrapped as clinical information.

Message Sequence	Description
1.2	esMD sends the synchronous request acknowledgment to the HIH.
2.1	The esMD system processes the provider information received in the service registration request, and the success confirmation or error(s) are returned for any validation failures as the first notification. esMD sends one of the following notifications (asynchronous) to the HIH after completing processing of the service registration request: 1. esMD - Request Accepted. 2. esMD - Request Accepted with Errors. 3. esMD - Meta Data Validation and Persistence.
2.2	The HIH acknowledges the acceptance/rejection status of the notification received from esMD.
3.1	esMD batches all the Service Registration requests and sends them to all of the MAC RCs.
3.2	esMD batches all the Service Registration requests and sends them to all of the Data Centers (DC).

22. Document Codes Processing Overview

Table 21: Document Code File Process Flow Steps describes the typical Document Codes flow interaction as shown in Figure 91: Document code file (DCF) Flow

The RC Client API shall download/pull the new DCF Flat File from the esMD Cloud and initially move the file to the Temp Folder. The RC Client shall continue processing with the header, body, and trailer validation. The RC Client shall move the DCF file to input folder send a Success pickup notification on successful validation of the file. The RC Client shall push the error pickup notification to esMD if there is any validation failure and delete the downloaded DCF Flat file from the downloaded folder and processing ends.

The following items are checked during schema validation:

- The length of the flat file lines should be within the limits as mentioned in Section 23.1 DCF File Format.
- The number of document code flat file lines present in flat file should be equal to number mentioned in same flat file trailer.
- The name of the DCF file should contain the proper content type code of 17.
- The Header and Trailer should start with pre-defined character.

23. Schema Definition and Sample Files

23.1 DCF File Format

Figure 91: Document code file (DCF) Flow

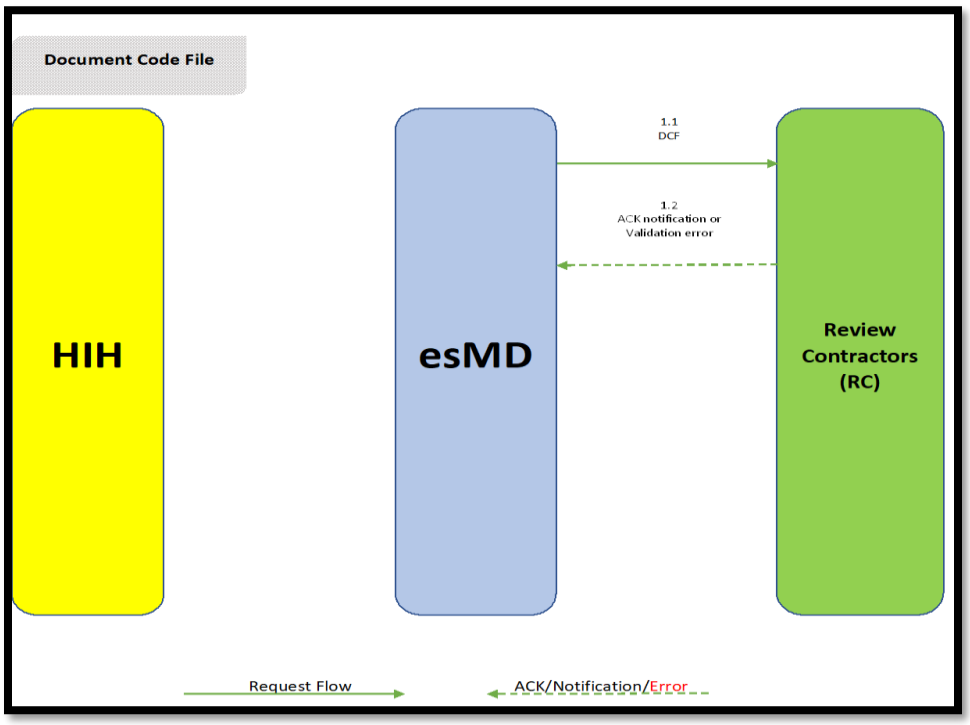


Table 21: Document Code File Process Flow Steps

Message Sequence	Description
1.1	The RC shall receive the DCF flat file from the esMD system via esMD Download API
1.2	The RC system processes the DCF flat file received from the esMD and generates the appropriate acceptance or rejection response acknowledgement to the esMD

23.2 eMDR (Pre-Pay/Post-Pay/Post-Pay-Other) Processing Overview

This section focuses on exchanging structured (Extensible Markup Language (XML)) and unstructured (Portable Document Format (PDF)) eMDR and ADR (Pre-Pay, Post-Pay and Post-Pay-Other) transactions in the form of electronic clinical documents and Nationwide Health Information Network (NwHIN)-Cross-Enterprise Document Reliable Interchange (XDR) profile standards, which may already exist in both the initiator and consumer entity systems or may need to be created for this exchange.

23.2.1 Logical Process Flow

23.2.1.1 eMDR Pre-Pay Logical Flow

The eMDR pre-pay logical flow depicts the series of events and sequence of interactions between esMD and Health Information Handlers (HIH) via the XDR interface. The order and timing of the exchange of messages with HIHs are driven by RC submissions. The Content Type Code for this program is changed from 1.5 to 2.5.

Figure 92: eMDR Pre-Pay Process Flow depicts the logical processing of the eMDR (Pre-Pay) process, and Table 22: eMDR Pre-Pay Logical Process Flow Steps details the eMDR process.

Figure 92: eMDR Pre-Pay Process Flow

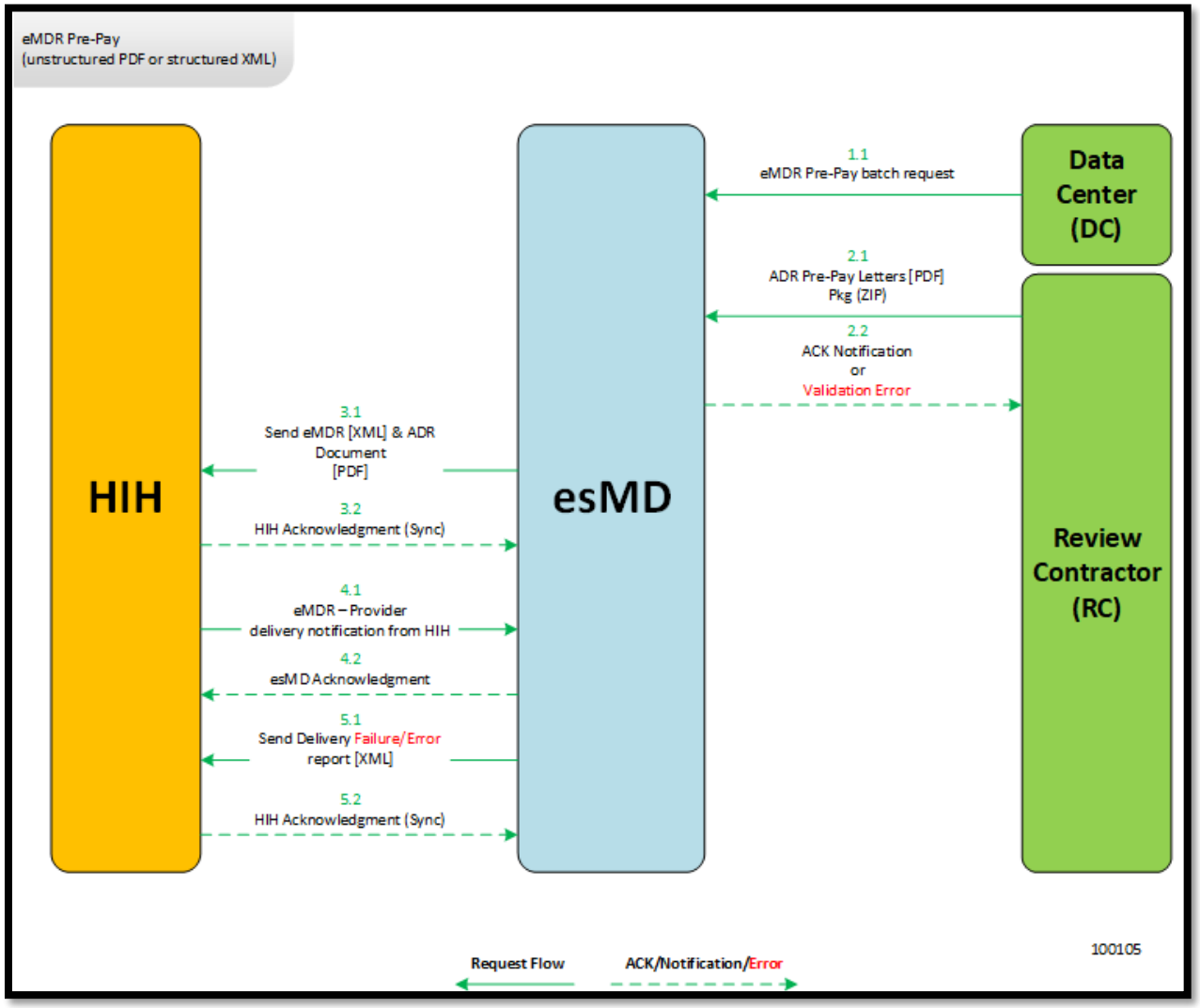


Table 22: eMDR Pre-Pay Logical Process Flow Steps

Message Sequence	Description
1.1	The esMD system receives the eMDR Pre-Pay batch request file from the Data Center (DC). The esMD system processes the eMDR (Pre-Pay) batch request file and holds the eMDR requests within the esMD system until the matching ADR letter (PDF) is received from the RC. The esMD system maintains the record of any processing errors or failures.
2.1	The RC sends ADR letters (PDF) matching the eMDR requests in the zip request package to esMD.
2.2	The esMD system processes the ADR letter zip packages received from the RC and generates the appropriate acceptance or rejection response acknowledgement to the RC. These messages are delivered to RC using esMD Status API.
3.1	The esMD system constructs the XDR request payload with the RC's ADR PDF letter and structured matching eMDR embedded in the unstructured HL7 clinical document standard and sends it to the HIH.
3.2	The HIH acknowledges the acceptance/failure with any of the following statuses for the document/request received from esMD: 1. RequestAccepted 2. ResponseAccepted 3. Success 4. Error
4.1	The HIH sends the package delivery confirmation to esMD after the ADR PDF letter and eMDR structured XML are successfully transmitted to the Provider.
4.2	esMD acknowledges the delivery confirmation received from the HIH.
5.1	The esMD system sends the transaction details only when HIH delivery failed due to validation error or transmission error.
5.2	The HIH acknowledges the acceptance/failure with any of the following statuses for the document/request received from esMD: 1. RequestAccepted 2. ResponseAccepted 3. Success 4. Error

23.2.2 eMDR Post-Pay/Post-Pay-Other Logical Flow

The eMDR Post-Pay/Post-Pay-Other logical flow depicts the series of events and sequence of interactions between esMD and the HIH via the XDR interface. The order and timing of the exchange of messages with HIHs are driven by RC submissions. The Content type code for this program is changed from 1.6 to 2.6.

Figure 93: eMDR Post-Pay/Post-Pay-Other Process Flow depicts the logical processing of eMDR (Post-Pay/Post-Pay-Other) process, and Table 23: eMDR Post-Pay/Post-Pay-Other Logical Process Flow Steps details the sequence of interactions between esMD and HIH.

Figure 93: eMDR Post-Pay/Post-Pay-Other Process Flow

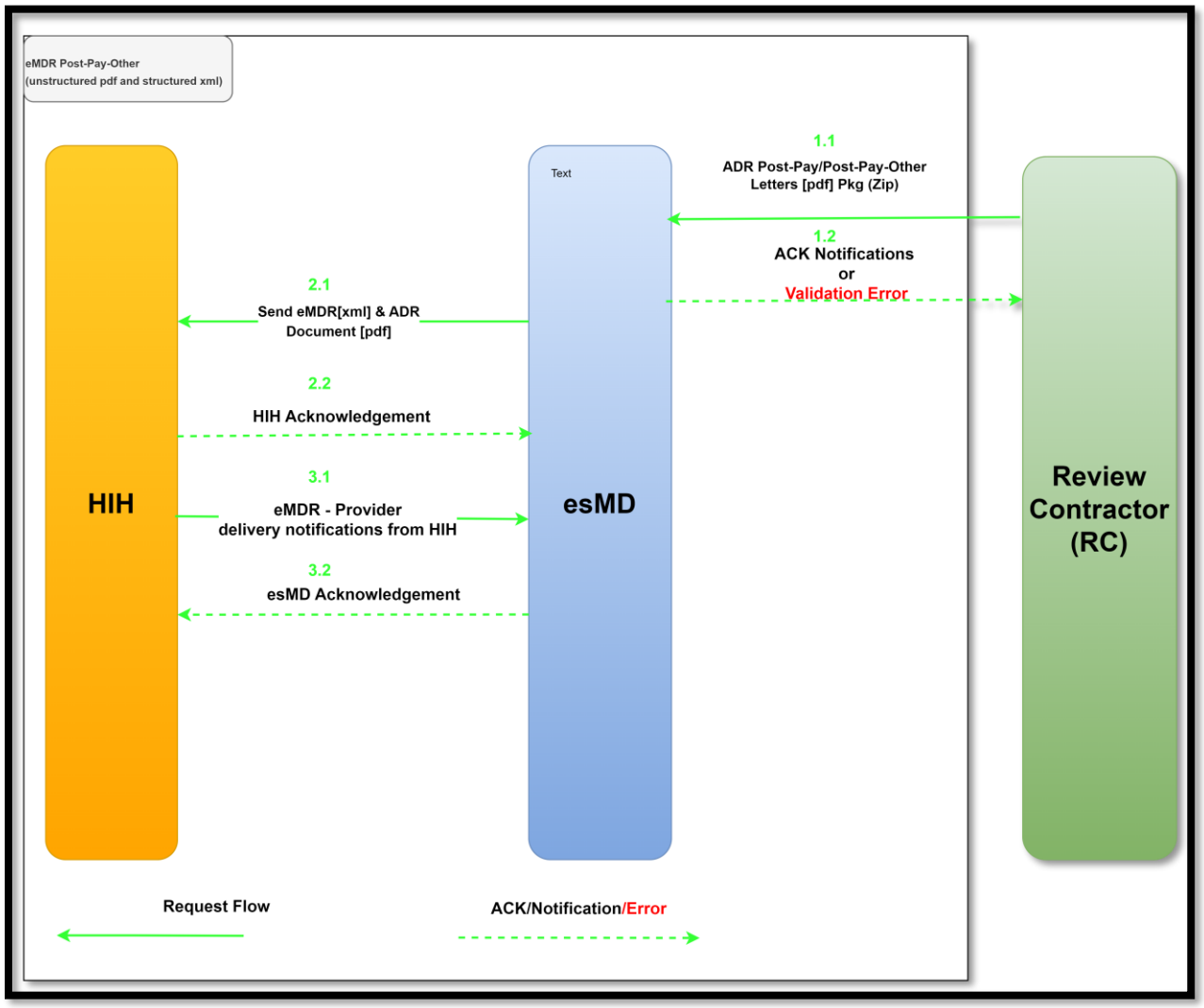


Table 23: eMDR Post-Pay/Post-Pay-Other Logical Process Flow Steps

Message Sequence	Description
1.1	The esMD system receives the ADR Letter PDF and structured eMDR Post-Pay /Post-Pay-Other XML in a zip file from the RC.
1.2	The esMD system processes the eMDR Post-Pay/Post-Pay-Other package received from the RC and generates the appropriate acceptance or rejection response acknowledgement to the RC. These messages are delivered to Review Contractor B using esMD Status API.
2.1	The esMD system constructs the XDR request payload with the ADR PDF letter and structured Post-Pay/Post-Pay-Other eMDR (XML) embedded in the unstructured HL7 clinical document standard and sends it to the HIH.

Message Sequence	Description
2.2	The HIH acknowledges the acceptance/failure with any of the following statuses for the document/request received from esMD: 1. RequestAccepted 2. ResponseAccepted 3. Success 4. Error
3.1	The HIH sends the package delivery confirmation to esMD after the ADR PDF letter and eMDR structured XML are successfully transmitted to the Provider.
3.2	The esMD system acknowledges the delivery confirmation received from the HIH.

As part of the October 2021 release, only one PDF letter can be submitted in the eMDR Post-Pay/Post-Pay-Other RC package. If more than one PDF letter is included in the RC package, the request will be rejected with an error message. In addition, the eMDR Post-Pay/Post-Pay-Other schema definition will be updated to remove the restriction for the Zip code under Sender Details.

23.3 Letters Logical Flow

The Letters logical flow depicts the series of events and sequence of interactions between RCs, esMD, and the HIHs via the XDR interface. The order and timing of the exchange of messages with HIHs are driven by RC submissions.

The description PADL/RRL is changed to LETTERS in all the places wherever applicable.

Figure 94: Letters Process Flow process, and Table 24: Letters Logical Process Flow Steps details the sequence of interaction between RC, esMD and HIH.

Figure 94: Letters Process Flow

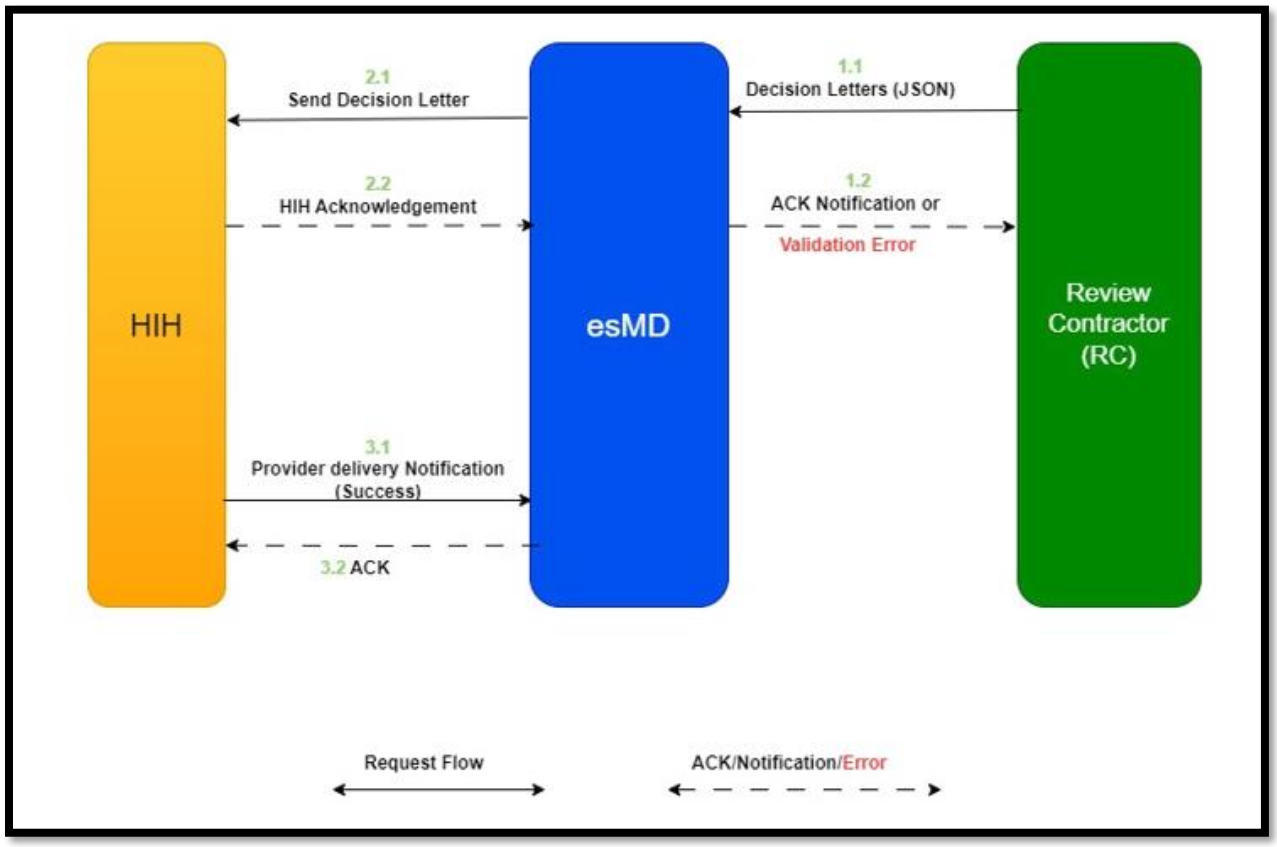


Table 24: Letters Logical Process Flow Steps

Message Sequence	Description
1.1	The esMD system receives the Letters JSON message from the RC.
1.2	The esMD system processes the Review Result Letter JSON message received from the RC and generates the appropriate acceptance or rejection response acknowledgement to the RC.
2.1	The esMD system constructs the Letters JSON message using the Structured Letter Record JSON message with the attached document embedded in the unstructured HL7 clinical document standard and sends it to the HIH.
2.2	The HIH acknowledges the acceptance/failure with any of the following statuses for the document/request received from esMD: 1. RequestAccepted 2. ResponseAccepted 3. Success 4. Error
3.1	The HIH sends the Letters Provider delivery confirmation to esMD after the Letter is successfully transmitted to the Provider.
3.2	The esMD system acknowledges the delivery confirmation received from the HIH.

23.4 Start RC Client

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

23.4.1 Login and Encryption

The RC Client prompts the user for the following details:

- IDM User ID
- IDM Password.

After successful login, the password is encoded in base64 format and then the login credentials (username and password) are encrypted in memory and used when required to authenticate by the Auth API for upload and download file operations from esMD Cloud. The RC Client initiates two threads, one for the inbound process and one for the outbound process. These processes are described on in sections 23.5 Upload Process and 23.6 Download Process, respectively.

23.5 Upload Process

23.5.1 Outbound Start

The esMD Auth API authenticates the RC Client user with their IDM credentials and returns a token which is valid for 15 minutes. The RC Client invokes the esMD Upload API with the token and other header information to get the pre-signed URL to upload the file. The RC Client loads the parameters for the outbound process from the RC Client XML configuration file. 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)
- UploadAPI URL and other details from properties file to upload the file to esMD Cloud
- Auth API details for the chosen environment (ESMDAuthAPI)

23.5.2 Get Outbound Documents

The RC Client checks the output directory for any files to be sent to the HIH. If any such files exist, the process continues and uploads the files to esMDotherwise, the outbound process thread sleeps for the time interval determined by the pushFrequency parameter in the XML Configuration file.

23.5.3 Connect

The RC Client connects to the esMD Cloud environment using IDM login credentials. The Encryption utility decrypts the credentials in memory and authenticates the user

using the esMD Auth API. If the user password is expired, the connection fails, and the user is prompted to provide the login information again.

23.5.4 Upload

The RC Client uploads the outbound files to the esMD Cloud. 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.

23.6 Download Process

23.6.1 Inbound Start

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

- Pull frequency
- Directories used by the RC Client to save the downloaded files (inputDirectory)
- DownloadAPI and Notification URL's and other details from properties file to upload the file to esMD Cloud and also to send notifications in real time.
- The ClientID and clientSecret used by the Auth API.

23.6.2 Housekeeping

The Housekeeping Manager is responsible for the cleanup and recovery from any abnormal terminations. If the extraction process was interrupted during extraction in the previous run, then there will be compressed files in the local "temp" directory.

23.6.3 Extraction

The Housekeeping Manager extracts compressed files found in the local "temp" directory for the RC Client before it downloads any new documents from the esMD Cloud. 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.

23.6.4 Checksum Verification

After the extraction is complete, the RC Client uses the XML Processor to parse the metadata file from the zip package. This 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.

Note: The cryptographic algorithm used to calculate and verify checksums is updated from SHA-1 to SHA-256 to make the application more secure. The updated

cryptographic algorithm is used to verify checksums for all inbound requests except for DCF and Service Registration, and to calculate checksums for outbound EMDR requests (Pre-Pay, Post-Pay, Post-Pay-Other), and ICDT requests.

23.7 Acknowledgements

23.7.1 Pickup Notification

If the RC Client successfully extracts and verifies the compressed files, the RC Client sends a SUCCESS notification by invoking the esMD Notification API to inform the HIH that the document has been received and successfully processed. The esMD Notification API is a synchronous call which receives the acknowledgement response back from the HIH and saves the XML file in the notification folder.

To generate this SUCCESS notification, the RC Client should:

1. Get the Transaction ID from the zip filename.
2. Prepare the notification with a SUCCESS message and generate an notification file.
3. Post the pickup notification message in JSON format to esMD by calling the esMD Notification API.
4. Convert the HIH Acknowledge JSON message received in the same synchronous call to XML format and save the file in the notification folder.

23.7.2 Error Pickup Notification

If the RC Client encounters an error indicating a failure while either extracting the compressed file or verifying the checksum for the contents of the package, the RC Client sends an error notification through the esMD system which asks the HIH to resubmit the package. To generate this error notification, the RC Client must:

1. Obtain the compressed file name.
2. Prepare the notification with an Error message.
3. Generate a JSON notification message.
4. Post the JSON pickup notification message to esMD by calling the esMD Notification API.

23.8 Housekeeping

After the Housekeeping Manager completes preprocessing, the RC Client authenticates the user by calling the Auth API with the encrypted username and password stored in memory. The RC Client also enables the User ID and Password fields to log in into the RC Client. Once the RC Client is logged in successfully, the processing will start.

23.9 Process Document

If any documents are available for the RC Client in the esMD Cloud, the RC Client will go through the list to pull each document using the esMD Download API and process the document.

23.10 Download Document

The RC Client uses the esMD Download API to pull each inbound document from esMD Cloud. The RC Client then extracts the contents of the zip file and continues processing.

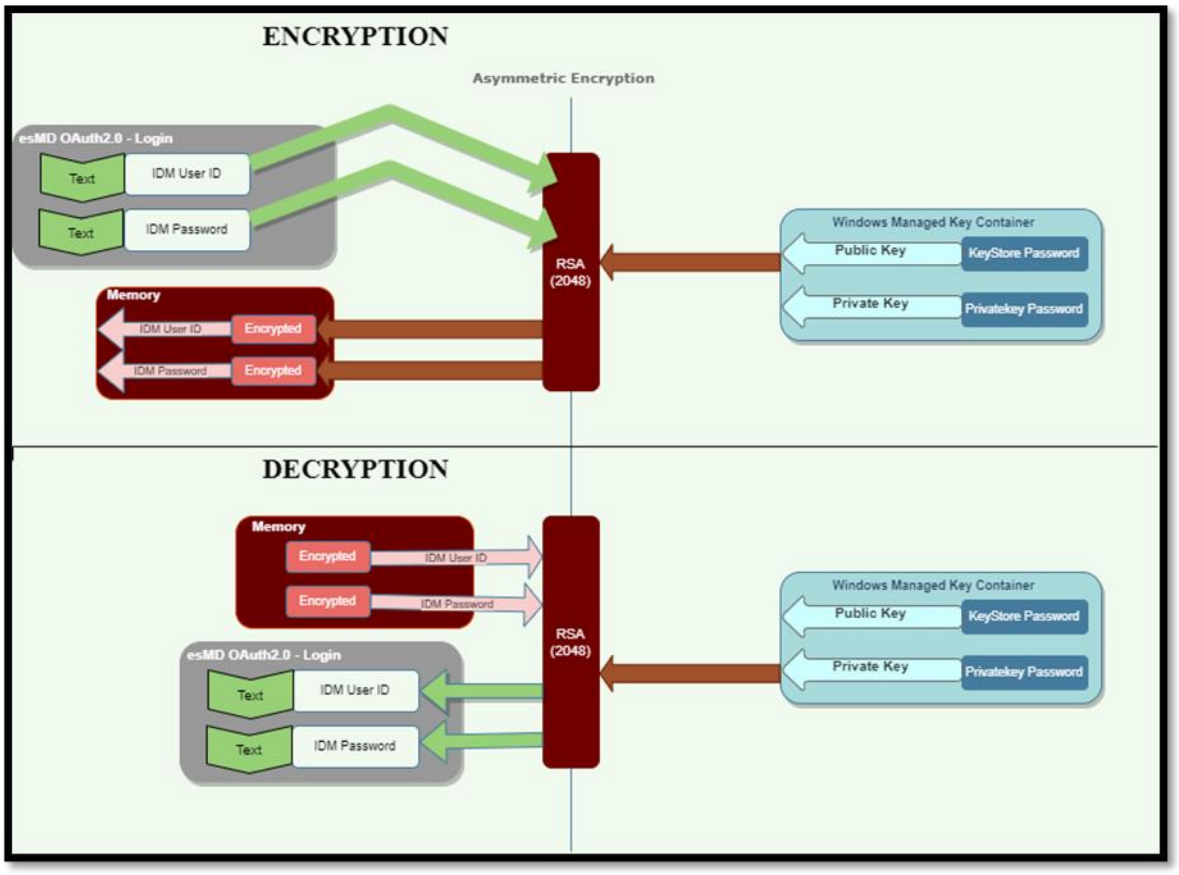
24. Java Client API

24.1 Security

When the RC Client starts, the user credentials are provided because they are stored in encrypted form in memory. Figure 95: Encryption and Decryption Process shows the processes used to safeguard the IDM user credentials from exposure.

The RC Java Client uses RSA asymmetric encryption algorithms to secure the login credentials.

Figure 95: Encryption and Decryption Process



24.2 Java API Documentation

This section discusses API methods that can be called for a custom solution to interface with the esMD system. This section may be skipped when choosing to use the RC Java client out-of-the-box.

24.2.1 Login

Table 25: Login Details shows the methods and their descriptions used in the login process.

Table 25: Login Details

No.	Method	Description
1.	public Login Details loginAndEncrypt(authAPIDetails , String xmlConfigFilename, ESMDConfig.KeyStoreInfo keyStoreInfo_, LoginBean loginBean_) throws Exception	Logs into the server and stores the encrypted login information. Parameters: 1. authAPIDetails – The AUTH API Details to connect to AUTH server for authentication. 2. xmlConfigFilename – Filename for XML Configuration details. 3. keyStoreInfo_ – The Keystore Details. 4. loginBean_ – The Login Details (User ID and Password). Returns: The LoginDetails Object with the following properties populated: 1. encryptedUID – Encrypted User ID. 2. encryptedPWD – Encrypted Password. 3. message – status(TRUE/FALSE) and description if any exceptions occurred.
2.	public LoginDetails decryptAndLogin(LoginDetails_, ESMDConfig.KeyStoreInfo keyStoreInfo_) throws Exception	Decrypts the login credentials passed in the LoginDetails object and logs into the TIBCO MFT server. Parameters: 1. loginDetails_ – the LoginDetails object with the following properties populated: • encryptedUID – Encrypted User ID. • encryptedPWD – Encrypted Password. 2. keyStoreInfo_ – The Keystore Details. Returns: The LoginDetails Object with the following properties populated: 1. encryptedUID – Encrypted User ID. 2. encryptedPWD – Encrypted Password. 3. channelSftp – SFTP Channel connected.

24.2.2 Download

Table 26: Inbound Method Details lists the methods and their descriptions used in the inbound process.

Table 26: Inbound Method Details

No.	Method	Description
1.	public List<String> getListOfFiles(ESMDConfig esmdConfig) throws Exception	Uses the LoginDetails object to list the remote directory. Parameters: 1. esmdConfig – RC Client configuration object Returns: The List<String> with the filenames to pull.
2.	Public static String getPresignedURL(ESMDConfig esmdConfig, Stirng key) throws Exception	Gets a presignedURL for the filename to access the file in the esMD Cloud . 1. esmdConfig – RC Client configuration object. 2. Key – zip filename
3.	public void downloadFileWithPresignedURL(String presignedURL, String localTempDirectoryPath) throws Exception	3. Pulls the document (namely, the zip file) from the esMD Cloud with the name presignedURL. 4. Saves it in the localTempDirectoryPath. Parameters: 1. presignedURL – The presignedURL of the file stored in the esMD Cloud. 2. localTempDirectoryPath – The local temporary file path to save the downloaded file.
4.	public String extractDocument(File localDocumentName_, File localTargetDirectory_) throws Exception	Extracts the zip file downloaded from the esMD Cloud. Parameters: 1. localDocumentName_ - The local zip file to extract. 2. localTargetDirectory _ - The target directory to place the extracted contents. Returns: The extracted Directory name as a String.

No.	Method	Description
5.	public boolean processMedicalDocumentation(String remoteDocumentName_)	This method does the following: 1. Extracts the zip file into the “download” directory using the extractDocument() method. 2. If extraction fails, calls the acknowledge method with an error event. 3. After successful extraction, verifies the extracted payloads against the checksum in the metadata file using the checkPayloads() method. 4. If checksum fails, calls the acknowledge method with an error event. 5. If checksum passes, calls the acknowledge() method with a success event. Parameter: 1. localDocumentPath_ - The local document name to process. Returns: The Boolean status of the processing for that document.
6.	public String acknowledge(RCPickupNotification, String randomNumber, String programType) throws Exception	Generates the pickup notification JSON message for a downloaded document. If the ErrorInfo object is populated, it generates an error pickup notification JSON message. If the ErrorInfo object is null, it generates a Success pickup notification. Parameter: 1. rcPickupNotification_ - The RCPickupNotification object. Returns: The Pickup Notification JSON string message.
7.	Public static String sendNotificationToESMD(ESMDConfig esmdConfig, String notification) throws Exception	This method posts the JSON notification message to esMD Notification API. 1. esmdConfig – ESMD Configuraiton details. 2. notification – Pickup Notification JSON message.
8.	Public static void createHIHDeliveryAckResponseXML(ESMDConfig esmdConfig, String ackResponse) throws JAXBException, IOException	Creates HIH Delivery Acknowledgement XML file with the input JSON response message. 1. esMDConfig – ESMD Configuration details. 2. ackResponse – Acknowledgement message from HIH in JSON format.
9.	Public static void createHIHPARejectResponseXML (ESMDConfig, String ackResponse)	Creates HIH Admin Error and PAReject Acknowledgement XML file with the input JSON response message. 1. esmdConfig – ESMD Configuration object. 2. ackResponse – Acknowledgement message from HIH in JSON format.

No.	Method	Description
10.	Public static void createICDTDeliveryAckResponseXML (ESMDConfig, String icdtAckResponse)	3. This method creates the ICDT pickup notification response xml from esMD and saves the xml in the ICDT notification folderesmdConfig – ESMD Configuration object. icdtAckResponse – Acknowledgement message from esMD in JSON format.
11.	Public static void createICDTErrorDeliveryAckResponseXML (ESMDConfig, String icdtErrorResponse)	4. This method creates the ICDT pickup notification Error response xml from esMD and saves the xml in the ICDT notification folderesmdConfig – ESMD Configuration object. icdtErrorResponse – Error Response message from esMD in JSON format.
12.	public boolean checkPayloads(File localExtractedDirectory_, RetrieveMedicalDocumentationResponse retrieveMedicalDocumentationResponse_)	Checks the payload against the metadata from the package. Parameters: 1. localExtractedDirectory_ – The directory in which the payloads, were extracted to as a File. 2. retrieveMedicalDocumentationResponse_ – The metadata xml as object. Returns: The status of the checksum verification. **Note**: The cryptographic algorithm used to calculate and verify the checksum is updated from SHA-1 to SHA-256 to make the application more secure.

24.2.3 Upload

Table 27: Retrieval of Outbound Documents Details lists the methods and their descriptions used in the outbound process.

Table 27: Retrieval of Outbound Documents Details

No.	Method	Description
1.	public static String getPresignedURL(ESMDConfig, String filePath) throws Exception	This method is used to get a presignedURL for the file to be uploaded to esMD Cloud. Parameters: esmdConfig – ESMD Configuration details.filePath_ –The full filepath to be uploaded.Returns: The String value of the presignedURL.

No.	Method	Description
2.	public static void uploadFileWithPresignedURL(String presignedURL, String outboundDirPath) throws Exception	This method is used upload a local compressed document from the “output” directory to the esMD Cloud with the presigned URL. Parameters: 1. presignedURL_ – The presignedURL of the file to be uploaded. 2. outboundDirPath_ – The outbound directory of the file to be uploaded.

24.2.4 Upload Realtime API

Table 28: Upload Realtime API Details

No.	Method	Description
1.	public Message submitLetters(LettersBean lettersbean, File lettersPDFFile) throws Exception	This method is used to upload the Letters JSON file to esMD. Parameters: 1. lettersbean – Letters Bean object. 2. lettersPDFFile – PDF File location. **Note: Updated the names of classes, interfaces, method names, folder names, file names from 'rrl' to 'letters' wherever applicable**

24.2.5 Status

Table 29: Status Method Details lists the methods and their descriptions used in the notification process.

Table 29: Status Method Details

No.	Methods	Description
1.	public static String getStatusFromESMD (ESMDConfig esmdConfig_)	This method takes esmdConfig object as input which has the configuration details from the xml file and invokes the esMDStatus API to get the status Parameter: 1. paErrorResponseBean_ – The PAErrorResponseBean object to use. Returns: The Message Object which has status of validations result and also the list of Validation Failure Bean object if there is any validation failure with the data provided by the user.
2.	public String createPAErrorResponseObject(PAErrorResponseBean paErrorResponseBean_) throws Exception	This method takes PAErrorResponseBean object as input which has the review error (rejected decision) response information provided by user and creates the PA Reject Response JSON message. Parameter: 1. paErrorResponseBean_ – The PAErrorResponseBean object to use. Returns: The PARejectResponse JSON String populated with the data provided by the user.

24.2.6 PA Error (Rejected Decision) Response

Table 30: Manual Submission of PA/PCR (Rejected Decision) Response details the methods to submit the PA and HHPCR Error (Rejected Decision) Response.

Table 30: Manual Submission of PA/PCR (Rejected Decision) Response

No.	Methods	Description
1.	public Message validationOfPAErrorResponse (PAErrorResponseBean paErrorResponseBean_)	This method takes PAErrorResponseBean object as input which has the review error (rejected decision) response information provided by user and validates all that information before generating the response XML. Parameter: 1. paErrorResponseBean_ – The PAErrorResponseBean object to use. Returns: The Message Object which has status of validations result and also the list of Validation Failure Bean object if there is any validation failure with the data provided by the user.
2.	public String createPAErrorResponseObject(PAErrorResponseBean paErrorResponseBean_) throws Exception	This method takes PAErrorResponseBean object as input which has the review error (rejected decision) response information provided by user and creates the PA Reject Response JSON message. Parameter: 1. paErrorResponseBean_ – The PAErrorResponseBean object to use. Returns: The PARejectResponse JSON String populated with the data provided by the user.

24.2.7 Administrative Error Response to Inbound Submissions

Table 31: Manual Submission of Administrative Error Response details the methods to submit the Administrative Error Response to an Inbound submission.

Table 31: Manual Submission of Administrative Error Response

No.	Methods	Description
1.	public Message validationOfAdministrativeErrorResponse (AdministrativeErrorResponseBean administrativeErrorResponseBean_)	This method takes AdministrativeErrorResponseBean object as input which has the administrative error response information provided by user and validates all that information before generating the response XML. Parameter: 1. administrativeErrorResponseBean_ – The AdministrativeErrorResponseBean object to use. Returns: The Message Object which has status of validations result and the list of Validation Failure Bean object if there is any validation failure with the data provided by the user.

No.	Methods	Description
2.	private String createAdministrativeErrorResponseObject(AdministrativeErrorResponseBean administrativeErrorResponseBean_) throws Exception	This method takes AdministrativeErrorResponseBean object as input which has the administrative error response information provided by user and creates the Administrative Error Response JSON String. Parameter: 1. administrativeErrorResponseBean_ – The AdministrativeErrorResponseBean object to use. Returns: The Administrative Error Response JSON message populated with the data provided by the user.

24.2.8 Utilities - Encryption

Note: The Java Client release from April 28, 2014 does not include the encryption of login credentials. This section depicts the planned design and is subject to change. This guide will be updated as required when the security implementation is completed.

Table 32: Encryption provides the details on the esMD RC Client Encryption methods.

Table 32: Encryption

No.	Methods	Description
1.	public String encryptKSPassword(String keyStorePassword_) throws Exception	This method encrypts the Keystore password so it can be stored in the configuration file. Parameter: 1. keyStorePassword_– The password to encrypt as a String. Returns: The Encrypted Keystore Password using “PBEWithMD5AndTripleDES.
2.	public String encryptPKPassword(String privateKeyPassword_) throws Exception	This method encrypts the Private Key password so it can be stored in the configuration file. Parameter: 1. privateKeyPassword_ – The password to encrypt as a String. Returns: The Encrypted Private Key Password using “PBEWithMD5AndTripleDES”.

No.	Methods	Description
3.	public Map<String, String> encryptCredentials(Map<String, String> loginInfo_, ESMDConfig.KeyStoreInfo keyStoreInfo_) throws Exception	This method encrypts the IDM login credentials using an RSA Public Key from the JKS Store. Parameters: 1. loginInfo_ - The Map<String, String> containing the Unique Identification Number (UID) and PWD as keys. 2. keyStoreInfo_ - The ESMDConfig.KeyStoreInfo object with the following details populated: • keyStoreLocation – The JKS Store to use as a String. • encKeyInfo – The Encrypted Keystore password to load the JKS as a String. • certAlias – The alias of the certificate to retrieve the public key as a String. Returns: The Map<String, String> of encrypted login credentials ENC_UID and ENC_PWD as keys.
4.	public Map<String, String> decryptCredentials(Map<String, String> encryptedLoginInfo_, ESMDConfig.KeyStoreInfo keyStoreInfo_) throws Exception	This method decrypts the IDM login credentials using an RSA Private Key from the JKS Store. Parameters: 1. encryptedLoginInfo_ - The Map<String, String> of encrypted login credentials ENC_UID and ENC_PWD as keys. 2. keyStoreInfo_ - The ESMDConfig.KeyStoreInfo object with the following details populated: • keyStoreLocation – The JKS Store to use as a String. • encKeyInfo - The Encrypted Keystore password to load the JKS as a String. • certAlias - The alias of the certificate to retrieve the public key as a String. • encKeyInfoExt - The Encrypted private key password to load the private key from the JKS Store as a String. Returns: The Map<String, String> containing the UID and PWD as keys.

24.2.9 Test Connection

Refer to Table 33: Remote Troubleshooting for details on the ExecuteHandshake method.

Table 33: Remote Troubleshooting

No.	Methods	Description
1.	public bool executeHandshake()	This sample method invokes a call to the AUTH API server to pass login information and authenticate the user to assist in remote troubleshooting. Returns: TRUE if handshake succeeded.

25. API Methods

25.1 Unique ID Rules and Format

The Unique ID is generated based on the following format in the RC Client for the following Pilot Programs:

1. ICDT Solicited Request – CTC 15.1
2. ICDT Solicited Response – CTC 15.2
3. ICDT Unsolicited Response – CTC 15.3

Refer to Table 34: Example Unique ID Rules & Format for more information.

Table 34: Example Unique ID Rules & Format

ID	Format	Example	Notes
1	L<CTC><deliveryType><15 CharUniqueID><routingID> <date><time>	L13STSESD0020131191203 070	1. CTC for Solicited Request – 151 (period in the CTC is removed in the Unique ID) 2. CTC for Solicited Response – 152 (period in the CTC is removed in the Unique ID) 3. CTC for Unsolicited Response – 153 (period in the CTC is removed in the Unique ID) 4. routingId -- RCs mailbox ID 5. DeliveryType -- Q 6. date -- Date in MMDDYY format 7. time - Time in HHMMSS0

25.1.1 Unique ID Generation

Table 35: Unique ID Generation API Methods describes the API methods available to generate the Unique ID.

Table 35: Unique ID Generation API Methods

No.	Class Name	Method	Description
1	ICDTUtils	Public static String randomAlphaNumericValue() throws Exception	This method is used to generate the 3-character alphanumeric value that is used as input for generating the Unique ID. Parameter: None Returns: The String Object which has the 3-character alphanumeric value.
2	ICDTUtils	public static String generateUniqueID(String randomAlphaNumericValue_, String date_, String timestamp_) throws Exception	This method is used to generate a Unique ID for each ICDT Request and ICDT Response sent from the RC. Parameter:

No.	Class Name	Method	Description
			1. randomAlphaNumericValue_ - The value created using the randomAlphaNumericValue() method. 2. Date_ - The current system date in MMddyy format. 3. Timestamp_ - The current system timestamp in HHmmss format. Returns: The String Object of Unique ID value for the ICDT Request and ICDT Response.

25.2 ICDT Request

Table 36: ICDT Request API Methods details the methods available to submit the ICDT Request by different Review Contractors.

Table 36: ICDT Request API Methods

No.	Class Name	Method	Description
1	ICDTRequestProcessorImpl	Message generateICDTPackage(ICDTMetadataBean_, boolean isUniqueIdCreate_)	This method is used to create the Request XML based on the bean object values. icdtMetadataBean_ - ICDTMetadataBean object values to generate the Request XML file isUniqueIdCreate_ - Boolean value (true or false) to denote if the Request ID is to be generated by API. True if the Request ID is to be generated by API and false if the RCs provide the Request ID to the API. Returns Message Object - The Message Object contains messages, status, list of errors and desc, randomNumber, Request ID, and filename.
2	ICDTRequestProcessorImpl	Message generateICDTPackage(String icdtSolicitedRequestXMLFileLocation_, boolean isUniqueIdCreate_, Collection<File> icdtAttachmentFiles_)	This method is used to generate the ICDT Request package based on the absolute path of the Request XML file. icdtSolicitedRequestXMLFileLocation_ - The absolute file path of the request XML. isUniqueIdCreate_ - Boolean value (true or false) to denote if the Request ID is to be generated by API. True if the Request ID is to be generated by API and false if the RCs provide the Request ID to the API. icdtAttachmentFiles_ - List of attachment files to be included in the Solicited response package. Returns Message Object - The Message Object contains message, status, list of errors and desc, randomNumber, Request ID, and filename

No.	Class Name	Method	Description
3	ICDTRequestProcessorImpl	Message generateICDTPackage(File icdtSolicitedRequestXMLFileObj_, boolean isUniqueIdCreate_, Collection<File> icdtAttachmentFiles_)	This method is used to create the Request XML based on the file object icdtSolicitedRequestXMLFileObj__ - The request XML file object passed by RCs isUniqueIdCreate_ - Boolean value (true or false) to denote if the Request ID is to be generated by API. True if the Request ID is to be generated by API and false if the RCs provide the Request ID to the API icdtAttachmentFiles_ - List of attachment files to be included in the request package Returns Message Object - The Message Object contains message status list of errors and desc, randomNumber, Request ID, and filename
4	ICDTRequestProcessorImpl	ICDTRequest readRequestXMLFile(String xmlFileNamewithAbsolutePath_)	This method is used to read the ICDT Request XML file received from the esMD system xmlFileNamewithAbsolutePath_ - The absolute path of the Request XML downloaded in the RC Client

25.3 ICDT Solicited Response

Table 37: ICDT Solicited Response API Methods details the methods available for sending the ICDT Response by RCs.

Table 37: ICDT Solicited Response API Methods

No.	Class Name	Method	Description
1	ICDTSolicitedResponseProcessorImpl	Message generateICDTPackage(ICDTMetadataBean_, boolean isUniqueIdCreate_)	This method is used to generate the Solicited Response package based on the metadata bean object. icdtMetadataBean_ - ICDTMetadataBean object values for generating the Solicited Response isUniqueIdCreate_ - Boolean value (true or false) to denote if the Response ID is to be generated by API. True if the Response ID is to be generated by API and false if the RCs provide the Request ID to the API Returns Message Object - The Message Object contains message, status, list of errors and desc, randomNumber, Response ID, and the filename

No.	Class Name	Method	Description
2	ICDTSolicitedResponseProcessorImpl	Message generateICDTPackage(String icdtSolicitedResponseXMLFileLocation_, boolean isUniqueIdCreate, Collection<File> icdtAttachmentFiles_)	This method is used to create the Response XML based on the absolute path of the file. icdtSolicitedResponseXMLFileLocation_ - The absolute file path of the response XML isUniqueIdCreate_ - Boolean value (true or false) to denote if the Response ID is to be generated by API. True if the Response ID is to be generated by API and false if the RCs provide the Request ID to the API icdtAttachmentFiles_ - List of attachment files to be included in the Solicited response package Returns Message Object - The Message Object contains message, status, list of errors and desc, randomNumber, Request ID, and filename
3	ICDTSolicitedResponseProcessorImpl	Message generateICDTPackage(File icdtSolicitedRequestXMLFileObj_, boolean isUniqueIdCreate_, Collection<File> icdtAttachmentFiles_)	This method is used to create the Response XML based on the file object icdtSolicitedRequestXMLFileObj__ - The response XML file object passed by RCs isUniqueIdCreate_ - Boolean value (true or false) to denote if the Response ID is to be generated by API. True if the Response ID is to be generated by API and false if the RCs provide the Request ID to the API icdtAttachmentFiles_ - List of attachment files to be included in the Solicited response package Returns Message Object - The Message Object contains message, status, list of errors and desc, randomNumber, Request ID and filename
4	ICDTSolicitedResponseProcessorImpl	ICDTResponse readSolicitedResponseXMLFile (String xmlFileNamewithAbsolutePath_)	This method is used to read the ICDT Solicited Response XML file received from the esMD system xmlFileNamewithAbsolutePath_ - The absolute path of the Solicited Response XML downloaded in the RC Client

25.4 ICDT Unsolicited Response

Table 38: ICDT Unsolicited Response API Methods details the methods available for sending ICDT Response by RCs.

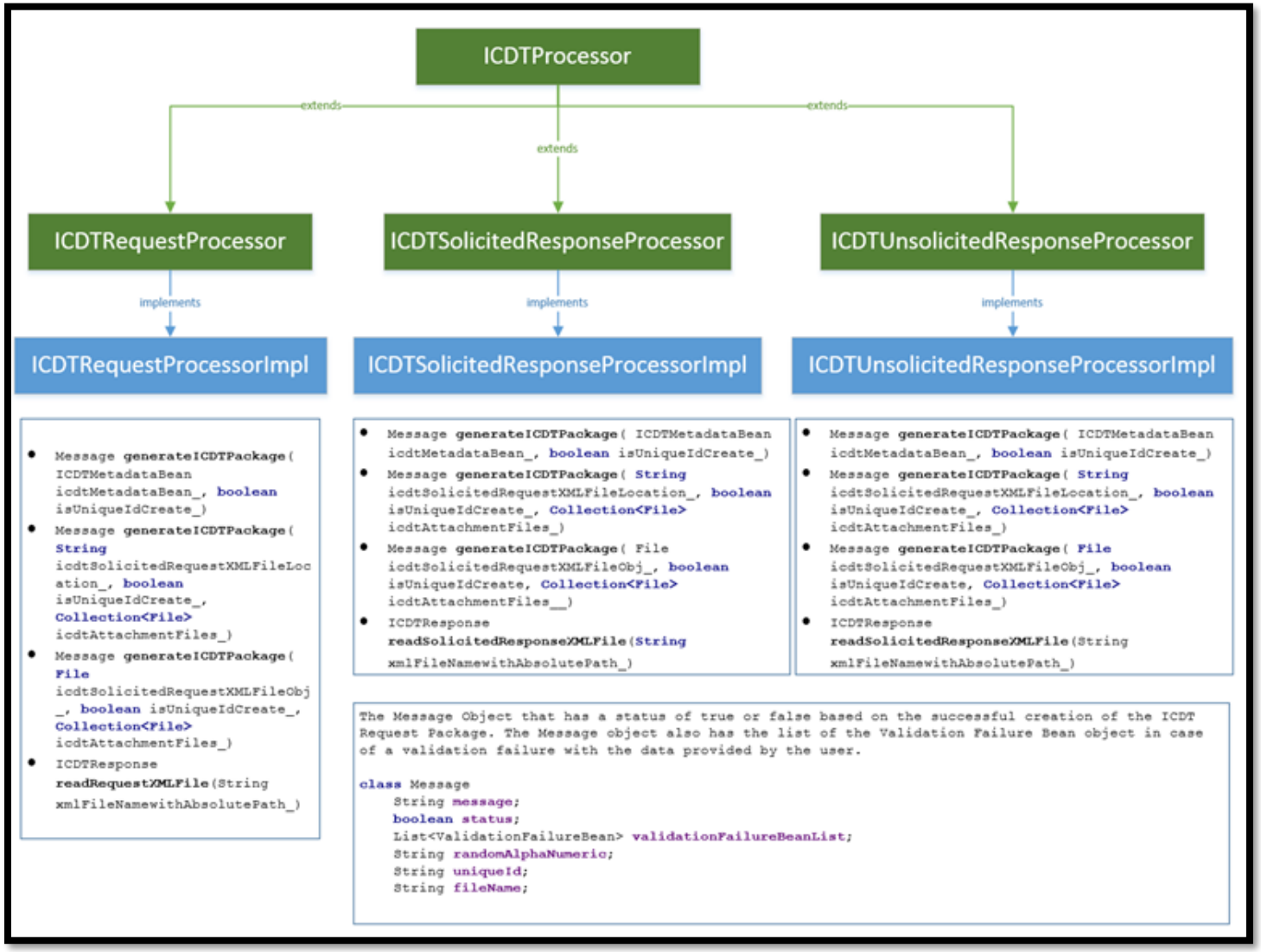
Table 38: ICDT Unsolicited Response API Methods

No.	Class Name	Method	Description
1	ICDTUnsolicitedResponseProcessorImpl	Message generateICDTPackage(ICDTMetadataBean_, boolean isUniqueIdCreate_)	This method is used to generate the Unsolicited Response package based on the metadata bean object. icdtMetadataBean_ - ICDTMetadataBean object values for generating the Unsolicited Response isUniqueIdCreate_ - Boolean value (true or false) to denote if the Response ID is to be generated by API. True if the Response ID is to be generated by API and false if the RCs provide the Request ID to the API Returns Message Object - The Message Object contains message, status, list of errors and desc, randomNumber, Response ID, and the filename
2	ICDTUnsolicitedResponseProcessorImpl	Message generateICDTPackage(String icdtUnsolicitedResponseXMLFileLocation_, boolean isUniqueIdCreate, Collection<File> icdtAttachmentFiles_)	This method is used to create the Request XML based on the absolute path of the file. icdtSolicitedResponseXMLFileLocation_ - The absolute file path of the Response XML isUniqueIdCreate_ - Boolean value (true or false) to denote if the Response ID is to be generated by API. True if the Response ID is to be generated by API and false if the RCs provide the Request ID to the API icdtAttachmentFiles_ - List of attachment files to be included in the UnSolicited response package Returns Message Object - The Message Object contains message, status, list of errors and desc, randomNumber, Request ID, and filename Note: RCs must pass dummy values for the file size tags.
3	ICDTSolicitedResponseProcessorImpl	Message generateICDTPackage(File icdtUnsolicitedRequestXMLFileObj_, boolean isUniqueIdCreate_, Collection<File> icdtAttachmentFiles_)	This method is used to create the Request XML based on the file object icdtUnsolicitedRequestXMLFileObj__ - The Response XML file object passed by RCs isUniqueIdCreate_ - Boolean value (true or false) to denote if the Response ID is to be generated by API. True if the Response ID is to be generated by API and false if the RCs provide the Request ID to the API icdtAttachmentFiles_ - List of attachment files to be included in the UnSolicited response package

No.	Class Name	Method	Description
			Returns Message Object - The Message Object contains message, status, list of errors and desc, randomNumber, Request ID, and filename Note: RCs must pass dummy values for the file size tags.
4	ICDTSolicitedResponseProcessorImpl	ICDTResponse readUnsolicitedResponseXMLFile (String xmlFileNamewithAbsolutePath_)	This method is used to read the ICDT Solicited Response XML file received from the esMD system xmlFileNamewithAbsolutePath_ - The absolute path of the Unsolicited Response XML downloaded in the RC Client

Figure 96: High-level ICDT API Architecture identifies the classes and method signatures for ICDT Request/Solicited Response and Unsolicited Response.

Figure 96: High-level ICDT API Architecture



25.5 Administrative Error Response

There are two additional Administrative error responses for supporting ICDT Request/Solicited Response and Unsolicited response:

1. The file is corrupt and/or cannot be read.
2. A virus was detected.

In order to generate Administrative error response for ICDT functionality, the API methods are provided with different method signatures as shown in Table 39: Administrative Error Response API Methods.

Table 39: Administrative Error Response API Methods

No.	Class/Interface Name	Methods	Description
1	ICDTAdminErrorNotificationProcessImpl	public Message generateICDTJSONNotification (NotificationBean adminErrorBean_) throws Exception	This method is used to generate the administrative error JSON response based on the bean object. adminErrorBean_ - AdminErrorBean object that holds the administrative error response details
2	ICDTAdminErrorNotificationProcessImpl	public ICDTCommunication readICDTNotification(String fileNameWithAbsolutePath_)throws Exception	This method is used to read the administrative error response based on the absolute path of the administrative error response file. fileNameWithAbsolutePath_ - The String object that has the absolute path of the administrative error response file.

25.6 Pre-Pay, Post-Pay, and Post-Pay-Other eMDR Letters

Table 40: Pre-Pay, Post-Pay, and Post-Pay-Other API Methods lists the API methods for generating the Pre-Pay, Post-Pay, and Post-Pay-Other eMDR letter packages.

Table 40: Pre-Pay, Post-Pay, and Post-Pay-Other API Methods

No.	Class/Interface Name	Methods	Description
1.	ESMDManualSubmitADReMDRPre-PayImpl	public Message generateEMDRPre-PayPackage(Collection<eMDR PrepayInfo> eMDRPrepayInfoList	Used to generate the ADR Pre-Pay letters zip package that contains the PDF file(s), eMDR process metadata XML file. Parameters: • eMDRPrepayInfo – The eMDRPrepayInfo bean object has the fileInfo and the corresponding NPI details. • eMDRPrepayLetterFiles – List of PDF Files. Returns: The Message that has a status of either True or False. The Message also has the list of validation failure objects in case of a validation failure, with the data provided by the user. Note: The existing method to generate the eMDR Prepay package is updated to support the new NPI metadata element addition. Existing method - public Message generateEMDRPrepayPackage(Collection<File> eMDRPrepayLetterFiles_, Message message)
2	ESMDManualSubmitADReMDRPostPayImpl	public Message generateEMDRPostpayPackage (Object eMDRPostPayStructuredBean_, Collection<File> eMDRPDFFiles_)	Used to generate the ADR Post-Pay letters zip package that contains the PDF file, structured XML bean, and PDF file. Parameters:

No.	Class/Interface Name	Methods	Description
			1. eMDRPostPayStructuredBean_ – The bean object of the structured XML file. 2. eMDRPDFFiles_ - The PDF file for the ADR letter file. Returns: The Message object that has a status of True or False based on the successful creation of the ADR letters zip package. The Message object also has the list of validation failure objects in case of a validation failure, with the data provided by the user.
3	ESMDManualSubmitADReMDRPostPayImpl	public Message generateEMDRPostpayPackage (File eMDRStructuredXMLFile_, Collection<File> eMDRPDFFiles_)	Used to generate the ADR Post-Pay letters zip package that contains the PDF file, structured XML file, and eMDR process metadata XML file. Parameters: 1. eMDRStructuredXMLFile_ – The absolute file path of the eMDR Structured file in XML format. 2. eMDRPDFFiles_ - The PDF file for the ADR letter file. Returns: The Message bean object that has a status of True or False based on the successful creation of the ADR letters zip package. The Message bean object also has the list of validation failure objects in case of a validation failure, with the data provided by the user.
4	ESMDManualSubmitADReMDRPostPayImpl	public Message generateEMDRPostpayPackage (string eMDRStructuredXMLFilePath_, Collection<String> eMDRPDFFilesPath_)	Used to generate the ADR Post-Pay letters zip package that contains the PDF file, structured XML file, and eMDR process metadata XML file. Parameters:

No.	Class/Interface Name	Methods	Description
			1. eMDRStructuredXMLFilePath_ – The absolute file path of the ADR letters in PDF format. 2. eMDRPDFFiles_ - The PDF file for the ADR letter file. Returns: The Message bean object that has a status of True or False based on the successful creation of the ADR letters zip package. The Message bean object also has the list of validation failure objects in case of a validation failure, with the data provided by the user.
5	ESMDManualSubmitADReMDRPostPayOtherImpl	public Message generateEMDRPostpayOtherPackage (Object eMDRPostPayOtherRequestBean_ , Collection<File> eMDRPDFFiles_)	Used to generate the ADR Postpay-Other letters zip package that contains the PDF file, structured XML bean, and PDF file. Parameters: 1. eMDRPostPayOtherRequestBean_ – The bean object of the structured XML file. 2. eMDRPDFFiles_ - The PDF file for the ADR letter file. Returns: The Message object that has a status of True or False based on the successful creation of the ADR letters zip package. The Message object also has the list of validation failure objects in case of a validation failure, with the data provided by the user.
6	ESMDManualSubmitADReMDRPostPayOtherImpl	public Message generateEMDRPostpayOtherPackage (File eMDRStructuredXMLFile_, Collection<File> eMDRPDFFiles_)	Used to generate the ADR Postpay-Other letters zip package that contains the PDF file, structured XML file, and eMDR process metadata XML file. Parameters:

No.	Class/Interface Name	Methods	Description
			1. eMDRStructuredXMLFile_ – The absolute file path of the eMDR Structured file in XML format. 2. eMDRPDFFiles_ - The PDF file for the ADR letter file. Returns: The Message bean object that has a status of True or False based on the successful creation of the ADR letters zip package. The Message bean object also has the list of validation failure objects in case of a validation failure, with the data provided by the user.
7	ESMDManualSubmitADReMDRPostPayOtherImpl	public Message generateEMDRPostpayOtherPackage (string eMDRStructuredXMLFilePath_, Collection<String> eMDRPDFFilesPath_)	Used to generate the ADR Postpay-Other letters zip package that contains the PDF file, structured XML file, and eMDR process metadata XML file. Parameters: 1. eMDRStructuredXMLFilePath_ – The absolute file path of the ADR letters in PDF format. 2. eMDRPDFFiles_ - The PDF file for the ADR letter file. Returns: The Message bean object that has a status of True or False based on the successful creation of the ADR letters zip package. The Message bean object also has the list of validation failure objects in case of a validation failure, with the data provided by the user.

25.7 Logs

Table 41: RC Client Logs lists the logs the RC Client provides. The RC is advised to monitor the logs for errors and exceptions.

Table 41: RC Client Logs

Log	Description
config.log	Logging for the encryptConfig.bat utility.
handshake.log	Logging for the test connection process.
rc.log	Logging for the sample application.
Inbound.log	Logging for the Inbound Process.
outbound.log	Logging for the Outbound Process.
response.log	Logging for the Response File (PA Review and Administrative Error) Creation Process.
request.log	Logging for the Request File (eMDR Request) Creation Process.

25.8 Utilities

Table 42: RC Client Utilities lists the utilities the RC Client provides.

Table 42: RC Client Utilities

Log	Description
encryptConfig.bat	Encrypts the provided passwords and updates the configuration XML file.
rcclient.bat	The RC Client User Interface Application. This application will have: 1) Login; 2) Error Response to PA Request; 3) Administrative Error Response to Inbound Submissions; and 4) Advanced/Debugging functionalities.

26. Error Codes

26.1 Errors: esMD to RC

Table 43: Error Codes Sent from esMD to RC lists all the error codes sent from the esMD to the RC.

Table 43: Error Codes Sent from esMD to RC

Error Code	Error Description
111	File not formatted correctly
118	ESMD validation error: Error encountered while saving ReviewContractorPickUpStatus data
144	Failure in sending the Administrative error response to HIH
145	Failure in sending the Administrative PA response to HIH
146	Failure in sending the Pickup notification to HIH
222	Number of Records Mismatch
303	esMD validation error: Empty file received in the response.
305	esMD validation error: Review Contractor PickUp Timestamp is not a valid Timestamp. Correct and resubmit.
306	esMD validation error: esMD Delivery Timestamp is not a valid Timestamp. Correct and resubmit.
333	Duplicate File Sent
501	Missing Contractor/Workload number
502	Missing esMD Transaction ID
503	Missing Mode of Receipt
504	Missing Service Trace Number
516	ESMD validation error: Error encountered while storing PA Review Results Response
517	ESMD validation error: Error encountered while fetching PA Review Results Response Notification objects
518	ESMD validation error: Error encountered while updating PA Review Results Response Notification objects to DB
534	Unzip error
535	Checksum error
536	Metadata error
537	Registration Request error
539	esMD internal system error (Unzip failure). Resubmit
541	esMD validation error: Transaction ID is invalid. Correct and resubmit

Error Code	Error Description
542	ESMD validation error: Outbound Content Type Code does not match Inbound Content Type Code for this transaction ID
543	ESMD validation error: RC is not authorized to use this Content Type Code
544	esMD validation error: Reason Code is required when Decision Indicator is N or R. Correct and resubmit.
545	esMD validation error: Total number of Reason Codes cannot exceed 25. Reduce the number of Reason Codes and Resubmit.
546	ESMD validation error: Warning: Total number of Denial Codes exceeds 25
547	ESMD validation error: Denial Code Description must be 2 - 256 positions
555	ESMD validation error: Content Type Code does not exist
556	esMD validation error: Decision Indicator must be A, N, M or R. Correct and resubmit.
557	esMD validation error: Review Contractor Unique Tracking Number must be 1 - 50 alphanumeric characters with no special characters. Correct and resubmit.
558	esMD validation error: Reason Code does not exist in the esMD database. Correct and resubmit.
559	ESMD validation error: Denial Code Description is required, if Decision Indicator is N or R and Denial Code is "Other."
560	esMD validation error: Submission is infected with virus. Correct and resubmit.
562	esMD validation error: Unique Tracking Number is required when Decision Indicator is A, N, or M. Correct and resubmit.
563	Error encountered while validating the PA Review Results Response
565	esMD Internal System Error: Unable to process your response. Correct and resubmit.
566	esMD validation error: A required element is either missing, has an invalid element format, or has an invalid length. Correct and resubmit.
567	esMD validation error: A Decision Indicator of 'M' is invalid for PMD PA or DMEPOS response. Provide a valid Decision Indicator and resubmit.
569	esMD validation error: Number of Approved Units, Approved Service Date, and Date Range are not allowed for this response. Correct and resubmit.
572	esMD validation error: Approved Service End Date is less than or equal to Approved Service Start Date. Correct and resubmit.
576	esMD validation error: Number of Approved Units, Approved Service Date, Approved Service Date Range, Industry Code(s) and Reason Code(s) are not allowed for this response. Correct and resubmit.
577	esMD validation error: Unable to parse response XML file. Correct XML and Resubmit.
600	esMD validation error: Duplicate Reason Codes found. Correct and resubmit.
601	Invalid Contractor/Workload Number

Error Code	Error Description
601	esMD validation error: Procedure Code in response not equal Procedure Code in request. Correct and resubmit.
602	esMD validation error: Approved Service Date must be greater than or equal to current system date.
603	esMD validation error: Decision Indicator = R; response is missing at least one combination of Error Category Code: Error Code. Add the combination(s) of Error Category Code: Error Code and Resubmit.
604	esMD validation error: More than 9 Error Codes were reported for a single Error Category Code. Reduce the number of errors for each Error Category Code to 9 and Resubmit.
605	esMD validation error: Decision Indicator = R; Category Code is invalid for the combination of Error Category Code: Error Code. Correct the Error Category Code and resubmit with correct combination(s) of Error Category Code: Error Code.
606	esMD validation error: Decision Indicator = R; invalid Error Code for the combination of Error Category Code: Error Code. Correct the Error Code and Resubmit with correct combination(s) of Error Category Code: Error Code
607	esMD validation error: Invalid Industry Code. Correct and resubmit.
608	esMD validation error: Invalid Reason Code. Correct and resubmit.
610	esMD validation error: Empty File Received in the Response. Correct and resubmit.
611	esMD validation error: Multiple Files Received in the Response. Resubmit with only one file
612	esMD validation error: Approved Service Date or Approved Service Date Range and Approved Unit are not allowed for this response. Correct and resubmit.
613	esMD validation error: Administrative error code is invalid. Correct and resubmit
614	esMD validation error: Approved Service End Date is less than the Current Date. Correct and resubmit
615	esMD validation error: Invalid error in the pickup notification. Correct and resubmit.
616	esMD validation error: Intended Recipient OID is deactivated and cannot accept response. Correct and resubmit.
617	esMD validation error: Mailbox ID in the response does not match with the Mailbox ID that the request was sent.
618	esMD validation error: Intended Recipient OID is deactivated and cannot accept response.
619	esMD validation error: Mailbox ID in the response does not match with the Mailbox ID that the request was sent.
620	esMD validation error: Invalid Review Response Creation Time format
621	esMD validation error: Invalid review Response Submission Time Format
622	esMD validation error: The Decision Indicator is not valid for this response. For a PA Response, it must be A, M or N. For an Error Response, it must be R. Correct and resubmit.

Error Code	Error Description
623	esMD validation error: Both Approved Service Date and Approved Service Date range cannot exists in same response. Correct and Resubmit.
624	esMD validation error: Approved Service Start Date cannot be greater than the Approved Service End Date. Correct and resubmit
625	esMD validation error: Reason Code is not allowed for Decision Indicator A. Correct and resubmit.
631	esMD validation error: A Review or Error Response is not allowed for this transaction.
632	esMD validation error: Total number of Industry Codes cannot exceed 5. Reduce the number of Industry Codes and resubmit.
633	esMD validation error: Either HIH is not active or agreement has expired to receive the response.
634	esMD validation error: Invalid Number of Approved Unit value, The Number of Approved Unit value should be greater than zero, a non-negative whole number.
637	esMD validation error: Outbound response received for the submission that failed for the Inbound
638	OUTBND_EMPTY_PICKUP_FILE_ERROR
640	esMD validation error: Intended recipient OID and Procedure Code is not a valid combination. Correct and resubmit
701	Missing esMD Transaction ID
702	Missing Procedure Code
703	Missing Decision Indicator
704	Missing Subscriber ID
705	Missing Workload Number
706	Missing Service Trace Number
800	ESMD validation error: Error occurred while storing the Review Contractor Status PickUp
801	Invalid esMD Transaction ID
801	ESMD validation error: Error occurred while validating the Review Contractor PickUp Status Data
802	Invalid Procedure Code
803	Invalid Decision Indicator
804	Invalid Subscriber ID
805	Invalid Workload Number
806	Invalid Service Trace Number
901	Invalid AAA codes
902	Invalid PA Program Reason Code
903	Invalid Review Decision Reason Code

Error Code	Error Description
904	esMD Validation Error: The ICDT Request Zip File received from RC is Zero Byte in size. Correct and resubmit.
905	esMD validation error: The Checksum received does not match the Checksum in the zip file. Correct and resubmit.
907	esMD validation error: The combination of Review Contractor OID and the Content type code received in the request from RC is incorrect. Correct and resubmit.
908	esMD validation error: The combination of HIH OID and the Content type code received in the eMDR request from RC is incorrect. Correct and resubmit.
909	esMD validation error: The Content type code received for the eMDR Request is incorrect. Correct and resubmit.
910	esMD Validation Error: The ICDT Request Zip File received from RC exceeded the maximum allowable size. Correct and resubmit.
911	esMD Validation Error: The Unique ID received in the eMDR Request Metadata XML File already exists in the database. Correct and resubmit.
912	esMD validation error: The name of the document does not match the document ID in the Metadata zip file. Correct and resubmit.
913	esMD validation error: The Size of Document received does not match with the Size of the Document in the zip file. Correct and resubmit.
914	esMD validation error: The Number of documents in the eMDR Request xml does not match with the number of documents in the zip file. Correct and resubmit.
915	esMD validation error: Unable to encode the response
916	esMD Validation Error: The eMDR request Zip file extraction failed. Correct and resubmit.
917	esMD validation error: Unable to parse request XML file. Correct XML and Resubmit.
918	esMD validation error: The Sender OID received from the Review Contractor for the eMDR Request is Invalid. Correct and resubmit.
919	esMD validation error: RC type provided in the metadata is Invalid for the eMDR request. Correct and resubmit.
920	esMD validation error: The name of the document does not match the document ID in the Metadata zip file. Correct and resubmit.
921	esMD validation error: The HIH OID received from the Review Contractor for the eMDR request is Invalid. Correct and resubmit.
922	esMD Validation Error: A Duplicate RC Unique ID received in the ADR Review Result Response XML File already exists. Correct and resubmit.

Error Code	Error Description
923	esMD Validation Error: The File received from RC exceeded the maximum allowable size for ADR Review Result Response. Correct and resubmit.
924	esMD Validation Error: The ADR Review Result Response Zip file extraction failed. Correct and resubmit.
925	esMD validation error: Unable to parse response XML file. Correct XML and Resubmit.
926	esMD validation error: The name of the document does not match the document ID in the Metadata zip file. Correct and resubmit.
927	esMD validation error: The Size of Document received does not match with the Size of the Document in the ADR Review Result Response zip file. Correct and resubmit.
928	esMD validation error: The Checksum received does not match the Checksum in the ADR Review Result Response zip file. Correct and resubmit.
929	esMD validation error: The HIH OID received from the Review Contractor for the ADR Review Result Response is Invalid. Correct and resubmit.
930	esMD validation error: The Sender OID received from the Review Contractor for the ADR Review Result Response is Invalid. Correct and resubmit.
931	esMD validation error: The Number of documents in the ADR Review Result Response does not match with the number of documents in the zip file. Correct and resubmit.
932	esMD validation error: The Content type code received for the ADR Review Result Response is incorrect. Correct and resubmit.
933	esMD validation error: The combination of Review Contractor OID and the Content type code received in the ADR Review Result Response from RC is incorrect. Correct and resubmit.
934	esMD validation error: The combination of HIH OID and the Content type code received in the ADR Review Result Response from RC is incorrect. Correct and resubmit.
935	esMD validation error: The MIME TYPE IS MISSING IN THE EMDR REQUEST Process Metadata. Correct and resubmit.
936	esMD Validation Error: The Document Unique ID received from RC for the eMDR Request exceeds the maximum length. Correct and resubmit.
937	esMD Validation Error: The Document Unique ID received from RC for the ADR Review Result Response exceeds the maximum length. Correct and resubmit.
938	esMD validation error: The MIME type is missing in the ADR Review Result Response Process Metadata. Correct and resubmit.
939	esMD validation error: Unable to parse {0} XML file. Correct XML and Resubmit
940	esMD Validation Error: The {0} received in the {1} XML File already exists in the database. Correct and resubmit

Error Code	Error Description
941	esMD validation error: The Receiver OID received from the Review Contractor for the {0} is non-participating. Correct and resubmit
942	esMD validation error: The Sender OID received from the Review Contractor for the {0} is non-participating. Correct and resubmit
943	esMD validation error: The Content type code received for the {0} is incorrect. Correct and resubmit
944	esMD validation error: The combination of Sender OID and the Content type code received in the {0} from RC is incorrect. Correct and resubmit
945	esMD validation error: The combination of Receiver OID and the Content type code received in the {0} from RC is incorrect. Correct and resubmit.
946	esMD Validation Error: The Claim ID received in the {0} is Invalid. Correct and Resubmit
947	esMD Validation Error: Missing Claim ID in the {0}. Correct and Resubmit
948	esMD Validation Error: The Case ID received in the {0} is Invalid. Correct and Resubmit.
949	esMD Validation Error: The NPI received in the {0} is Invalid. Correct and Resubmit
950	esMD Validation Error: Missing NPI in the {0}. Correct and Resubmit
951	esMD Validation Error: The HICN received in the {0} is Invalid. Correct and Resubmit
952	esMD Validation Error: Missing HICN in the {0}. Correct and Resubmit
953	esMD Validation Error: The OCN received in the {0} is Invalid. Correct and Resubmit
954	esMD Validation Error: Missing OCN in the {0}. Correct and Resubmit
955	esMD Validation Error: Sender OID and Receiver OID received in the {0} match. Correct and Resubmit
956	esMD Validation Error: Internal System issue
957	esMD validation error: The Checksum received does not match the Checksum in the Zip file. Correct and resubmit
958	esMD validation error: The MIME type is missing in the {0} Metadata. Correct and resubmit.
959	esMD validation error: The Size of Document received does not match with the Size of the Document in the {0} Zip file. Correct and resubmit
960	esMD Validation error: The number of documents received does not match the Number of Documents as stated in the {0} zip file. Correct and resubmit
961	esMD Validation error: The Mime type {0} is invalid. Correct and resubmit
962	esMD Validation Error: Missing NPI in the ICDT Request. Correct and Resubmit.
963	esMD Validation error: The name of the document does not match the name of the document received in the {0} in the zip file
964	esMD Validation error: The Request ID provided in the {0} is either missing or not exist in the esMD database
965	esMD Validation Error: Invalid Admin Error Code received from the Review Contractor

Error Code	Error Description
966	esMD Validation Error: The ICDT Request Zip File received from RC exceeded the maximum allowable size. Correct and resubmit
967	esMD Validation Error: The ICDT Request Zip File received from RC is Zero Byte in size. Correct and resubmit
968	esMD Validation Error: The {0} Zip file extraction failed. Correct and resubmit
969	esMD Validation Error: The documentation type received in the {0} XML is invalid. Correct and Resubmit.
970	esMD validation error: The MIME type is missing in the esMD Process Metadata. Correct and resubmit.
971	esMD Validation Error: The Document Unique ID received from RC for the ADR Review Result Letter exceeds the maximum length. Correct and resubmit.
972	esMD validation error: The Number of documents in the esMD Process Metadata xml does not match with the number of documents in the zip file. Correct and resubmit.
1032	esMD Validation Error: esMD system missed or received more than one {0} from the RC for the {1}. Correct and resubmit
1034	esMD Validation Error: Unable to parse the eMDR Process Metadata XML File for {0}. Correct and resubmit.
1035	esMD Validation Error: The Unique Id {0} received in the eMDR Process Metadata XML File for the {1} already exists in the database. Correct and resubmit.
1036	esMD Validation Error: The number of documents in the eMDR Process Metadata XML File does not match the number of documents in the zip file for the {0}.
1037	esMD Validation Error: The Checksum received does not match the Checksum calculated for one or more attachments in the zip file for the {0}. Correct and resubmit.
1038	esMD Validation Error: The name of the document for one or more attachments in the zip file does not match the name of the document in the {0} for the {1}. Correct and resubmit.
1039	esMD Validation Error: The size of the document for one or more attachments in the zip file does not match the size of the document received in the Metadata for the {0}. Correct and resubmit.
1040	esMD validation error: The Sender OID received from the RC for the {0} is invalid. Correct and resubmit.
1041	esMD Validation Error: The combination of Review Contractor OID and Content type code received for the {0} from RC is incorrect. Correct and resubmit.
1049	esMD Validation Error: Date of Service (To) is less then Date of Service (From) in the {0} for the {1}. Correct and resubmit.
1050	esMD Validation Error: The NPI {0} received from the Review Contractor is missing provider consent.
1051	esMD Validation Error: The NPI {0} received from the Review Contractor is not associated with any HIH in esMD. Correct and resubmit.
1052	esMD Validation Error: The combination of HIH OID and Content type code received for the {0} from RC is incorrect. Correct and resubmit.

Error Code	Error Description
1053	esMD Validation Error:Prefix for the {0} File Name does not match with the {1} in the {2} for the {3}. Correct and resubmit.
1054	esMD Validation Error: Invalid File extension for {0} received from RC for the {1}. Correct and resubmit.
1055	RC Client API Validation Error: {0} file size must be greater than {1}
1056	RC Client API Validation Error: {0} file size must not exceed {1}
1057	RC Client API Validation Error: Unable to parse the {0} for the {1}
1058	esMD Validation Error: The combination of HIH OID and Content type code received for the {0} from RC is incorrect. Correct and resubmit.
1059	The file attached to ADR esMDR Letter File received from Review Contractor is not a PDF or XML file
1063	esMD Validation Error: Either The Unique Letter ID Is Invalid OR Missing. Correct and resubmit.
1063	esMD Validation error: EITHER THE UNIQUE LETTER ID IS INVALID OR MISSING
1064	esMD Validation Error: The NPI {0} received from the Review Contractor is not associated with any HIH in esMD. Correct and resubmit.
1065	esMD Validation Error: Invalid format of the Document Unique ID provided in the eMDR Process Metadata XML File. Correct and resubmit.
1066	esMD validation error: The Sender OID received from the RC for the {0} is invalid. Correct and resubmit.
1068	esMD Validation Error: The MIME type is either missing or invalid in the ADR esMD Letters File Metadata XML File. Correct and resubmit.
1069	esMD Validation Error: The Document Unique ID received from the RC for the ADR esMD Letters File exceeds the maximum length. Correct and resubmit.
1070	esMD Validation Error: NPPES gateway response time out received for {0}. Please resubmit.
1071	esMD Validation error: EITHER THE TYPE OF EMDR IS INVALID OR MISSING
1072	esMD Validation error: EITHER THE ANALYSIS ID IS INVALID OR MISSING
1073	esMD Validation error: EITHER THE LETTER DATE IS INVALID OR MISSING
1074	esMD Validation error: EITHER THE ORGANIZATION NAME/RC DETAILS IS INVALID OR MISSING
1075	esMD Validation error: EITHER THE RC ADDRESS 1 IS INVALID OR MISSING
1076	esMD Validation error: EITHER THE RC CITY IS INVALID OR MISSING
1077	esMD Validation error: EITHER THE RC STATE IS INVALID OR MISSING
1078	esMD Validation error: EITHER THE RC ZIP CODE IS INVALID OR MISSING
1079	esMD Validation error: EITHER THE SENDER OR ORGANIZATION NAME IS INVALID OR MISSING
1080	esMD Validation error: EITHER THE PROVIDER LAST NAME OR ORGANIZATION NAME IS INVALID OR MISSING
1081	esMD Validation error: EITHER THE PROVIDER ADDRESS 1 IS INVALID OR MISSING

Error Code	Error Description
1082	esMD Validation error: EITHER THE PROVIDER CITY IS INVALID OR MISSING
1083	esMD Validation error: EITHER THE PROVIDER STATE IS INVALID OR MISSING
1084	esMD Validation error: EITHER THE PROVIDER ZIP CODE IS INVALID OR MISSING
1085	esMD Validation error: EITHER THE PROVIDER NPI IS INVALID OR MISSING
1086	esMD Validation error: EITHER THE RESPONSE DATE IS INVALID OR MISSING
1087	esMD Validation error: EITHER THE JURISDICTION OR ZONE INVALID OR MISSING
1088	esMD Validation error: EITHER THE PROGRAM NAME IS INVALID OR MISSING
1089	esMD Validation error: EITHER THE DOCUMENT CODE IS MISSING OR THE FORMAT IS INVALID
1090	esMD Validation error: EITHER THE CLAIM ID IS INVALID OR MISSING
1091	esMD Validation error: EITHER THE BENEFICIARY ID IS INVALID OR MISSING
1092	esMD Validation error: EITHER THE BENEFICIARY NAME IS INVALID OR MISSING
1094	esMD Validation Error: More than 5 Procedure Codes are received for a single error description. Maximum 5 Procedure codes are allowed. Correct and resubmit.
1095	esMD Validation Error: Procedure Code received in the RC Reject Response is not of a valid length and format. Correct and resubmit.
1096	esMD Validation Error: Procedure Code is missing in the RC Reject Response. Correct and resubmit.
1098	esMD Validation Error: Duplicate Procedure Codes received in RC Reject response. Correct and resubmit.
1100	esMD Validation Error: Notification Type is not a valid type. Correct and resubmit.
1210	esMD Validation Error: The Payload size received for the {0} exceeds the defined max payload size of 200MB. Correct and resubmit.
1211	esMD Validation Error: Validation Failed for {0}. Correct and resubmit with values in all required fields.
1212	esMD Validation Error: LETTER ID does not match with the Letter ID {0} in the {JSON} for the {1}. Correct and resubmit.
1213	esMD validation error: The MIME type is either Null or empty in the Document_Info Section for {0}. Correct and resubmit.
1214	EPOR Virus Scanning system is not responded in {0} minutes, esMD system timed out and failed to process. Resubmit the request.
1215	{0} Payload is infected with a Virus. Correct and Resubmit.
1216	esMD Validation Error: The LETTER ID {0} received in the JSON for the {1} already exists in the database. Correct and resubmit.
1218	esMD validation error: The MIME type is Invalid in the Document_Info Section for {0}. Correct and resubmit.
1219	Invalid Checksum. Correct and Resubmit.
1220	esMD Validation Error:Sender Routing ID is invalid in the received in {0} request. Correct and resubmit.

Error Code	Error Description
1221	esMD Validation Error: The Content type code received for the {0} is incorrect. Correct and resubmit.
1222	esMD Validation Error: Encoded Document is missing in the received in {0} request. Correct and resubmit.
1223	Virus Scanning system is not responded in {0} minutes, esMD system timed out and failed to process. Resubmit the request.
1245	esMD validation error: Reject reason code {0) submitted in Requester is invalid.
1246	esMD validation error: Reject reason code is required when reject reason is present in Requester level.
1247	esMD validation error: Reject Reason is required when reject reason code is present in Requester Level.
1248	esMD validation error: Reject reason code {0} submitted in Beneficiary is invalid.
1249	esMD validation error: Reject reason code is required when reject reason is present in Beneficiary level
1250	esMD validation error: Reject Reason is required when reject reason code is present in Beneficiary level.
1251	esMD validation error: Reject reason code {0} submitted in Patient Event is invalid.
1252	esMD validation error: Reject reason code is required when reject reason is present in Patient Event level.
1253	esMD validation error: Reject Reason is required when reject reason code is present in Patient Event level.
1254	esMD validation error: Reject reason code {0) submitted in Facility Provider is invalid.
1255	esMD validation error: Reject reason code is required when reject reason is present in Facility Provider level.
1256	esMD validation error: Reject Reason is required when reject reason code is present in Facility Provider Level.
1257	esMD validation error: Qualifier DK is required Ordering Provider.
1258	esMD validation error: Reject reason code {0} submitted in Ordering provider is invalid.
1259	esMD validation error: Reject reason code is required when reject reason is present in Ordering Provider level.
1260	esMD validation error: Reject Reason is required when reject reason code is present in Ordering Provider Level.
1261	esMD validation error: Qualifier SJ is required for Rendering or Supplier Provider.
1262	esMD validation error: Reject reason code {0} submitted in Rendering or Supplier provider is invalid.
1263	esMD validation error: Reject reason code is required when reject reason is present in Rendering Or Supplier Provider level.
1264	esMD validation error: Reject Reason is required when reject reason code is present in Rendering or Supplier Provider Level.
1265	esMD validation error: Qualifier DN is required for Referring Provider.
1266	esMD validation error: Reject reason code {0} submitted in Referring provider is invalid.
1267	esMD validation error: Reject reason code is required when reject reason is present in Referring Provider level.

Error Code	Error Description
1268	esMD validation error: Reject Reason is required when reject reason code is present in Referring Provider Level.
1269	esMD validation error: Qualifier 72 is required for Operating Provider.
1270	esMD validation error: Reject reason code {0} submitted in Operating provider is invalid.
1271	esMD validation error: Reject reason code is required when reject reason is present in Operating Provider level.
1272	esMD validation error: Reject Reason is required when reject reason code is present in Operating Provider Level.
1273	esMD validation error: Qualifier 71 is required for Attending Provider.
1274	esMD validation error: Reject reason code {0} submitted in Attending provider is invalid.
1275	esMD validation error: Reject reason code is required when reject reason is present in Attending Provider level.
1276	esMD validation error: Reject Reason is required when reject reason code is present in Attending Provider Level.
1277	esMD validation error: Reject reason code {0} submitted in Service is invalid.
1278	esMD validation error: Reject reason code is required when reject reason is present in Service level.
1279	esMD validation error: Reject Reason is required when reject reason code is present in Service Level.
1280	esMD validation error: Program reason code has invalid format.
1281	esMD validation error: Qualifier FA is required for Facility Provider.
1283	esMD validation error: Service line number is missing in Service level.
1284	esMD validation error: Invalid Service line number format in Service level.
1285	esMD validation error: Duplicate {0} Service Line number in Service level.
1286	esMD validation error: Invalid Service line number {0} in Service level.
1288	esMD validation error: Max 9 reject reason codes are allowed in {0} level.
LETTERS_SCHEMA_000	esMD validation error:{0} Ex: LetterId is missing, Category is missing, Provider last name or Organization name is missing etc..
LETTERS_001	esMD Validation error: Letter Id exceeds more than 60 characters in letters. Correct and resubmit.
LETTERS_002	esMD Validation error: Date of the Letter is invalid in letters. Correct and resubmit.
LETTERS_002	esMD Validation error: Unique letter Id exceeds more than 60 characters in letters. Correct and resubmit.
LETTERS_003	esMD Validation error: Category code is invalid. Correct and resubmit.
LETTERS_004	esMD Validation error: Sub Category code is invalid. Correct and resubmit.
LETTERS_005	esMD Validation error: Rc type exceeds more than 14 characters in letters. Correct and resubmit.

Error Code	Error Description
LETTERS_006	esMD Validation error: Jurisdiction /Region /Area/ Zone of the RC exceeds more than 40 characters in letters. Correct and resubmit.
LETTERS_007	esMD Validation error: Name of the review contractor exceeds more than 60 characters in letters. Correct and resubmit.
LETTERS_008	esMD Validation error: Line of Business exceeds more than 10 characters in letters. Correct and resubmit.
LETTERS_009	esMD Validation Error: Provider NPI is invalid
LETTERS_010	esMD Validation Error: Provider last name or Organization name exceeds maximum allowable of `00 characters in Letters
LETTERS_011	esMD Validation error: Provider Address 1 exceeds maximum allowable of 75 characters in Letters. Correct and resubmit.
LETTERS_012	esMD Validation error: Decision rationale exceeds maximum allowable of 5000 characters in Letters. Correct and resubmit.
1230	No Matching eMDR PDF file found for the Unique letter Id {0} and NPI {1} received in the eMDR Flat file
1231	No Matching Flat file found for the Unique letter Id {0} and NPI {1} received in the eMDR Process metadata file
E0001	expected type: JSON Object, found: Null
E0002	error : object has missing required properties ([\"notification\"])
E0002	error : object has missing required properties ([\"notificationType\"])
E0002	error : object has missing required properties ([\"esMDTransactionId\"])
E0002	error : object has missing required properties ([\"senderRoutingId\"])
E0002	error : object instance has properties which are not allowed by the schema: [\"senderroutingid1\"]

Note: The dynamic values {0} and {1}. will be replaced by the corresponding ID's and LOB values.

26.2 Errors: RC to esMD

There are two types of Error Codes sent by the RC to the esMD. They are:

1. Administrative Errors.
2. Pickup Errors.
3. PA Reject Errors.

26.2.1 Administrative Errors

Table 44: Administrative Error Codes lists the error codes used to report unexpected errors related to the payload received in a downloaded file from esMD. For more details, please refer to section 11.2.4 Administrative Error Response to Inbound Submissions.

Note: If the error name is "Other," an error description is mandatory; remaining admin error descriptions are optional.

Table 44: Administrative Error Codes

Administrative Error	Error Code	Description
Cannot Read Files/Corrupt Files	ESMD_410	ESMD_410- Administrative Error (Cannot Read Files/Corrupt Files).
Submission Sent to Incorrect RC	ESMD_411	ESMD_411- Administrative Error (Submission Sent to Incorrect RC).
Virus Found	ESMD_412	Submission is infected with a virus. This submission will not be processed by esMD. Resubmit new documentation.
Other	ESMD_413	ESMD_413- Administrative Error (Other).
Incomplete File	ESMD_414	ESMD_414- Administrative Error (Incomplete File).
Unsolicited Response	ESMD_415	ESMD_415- Administrative Error (Unsolicited Response).
Documentation cannot be matched to a case/claim	ESMD_416	ESMD_416- Administrative Error (Documentation cannot be matched to a case/claim).
Duplicate	ESMD_417	ESMD_417- Administrative Error (Duplicate).
The date(s) of service on the cover sheet received is missing or invalid.	GEX10	The date(s) of service on the cover sheet received is missing or invalid.
The NPI on the cover sheet received is missing or invalid.	GEX11	The NPI on the cover sheet received is missing or invalid.
The state where services were provided is missing or invalid on the cover sheet received.	GEX12	The state where services were provided is missing or invalid on the cover sheet received.
The Medicare ID on the cover sheet received is missing or invalid.	GEX13	The Medicare ID on the cover sheet received is missing or invalid.
The billed amount on the cover sheet received is missing or invalid.	GEX14	The billed amount on the cover sheet received is missing or invalid.
The contact phone number on the cover sheet received is missing or invalid.	GEX15	The contact phone number on the cover sheet received is missing or invalid.

Administrative Error	Error Code	Description
The beneficiary's name on the cover sheet received is missing or invalid	GEX16	The beneficiary's name on the cover sheet received is missing or invalid.
The claim number on the cover sheet received is missing or invalid.	GEX17	The claim number on the cover sheet received is missing or invalid.
The ACN on the coversheet received is missing or invalid.	GEX18	The ACN on the coversheet received is missing or invalid.

26.2.2 Pickup Errors

Table 45: Pickup Error Codes lists the types of error codes and their descriptions. These codes are used to populate the ErrorInfo object inside the error pickup notification. Refer to section 11.2.2 Error Pickup Notification for more details.

Table 45: Pickup 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.
ERROR DOCUMENT CODES VALIDATE FILE	515	Invalid line length for line 1; Expected: 1035, Actual: 1021 Invalid line length for line 2; Expected: 1035, Actual: 1028 Invalid line length for line 7; Expected: 1035, Actual: 1029 Invalid line length for line 8; Expected: 1035, Actual: 1029 Invalid line length for line 9; Expected: 1035, Actual: 1025 Note: This is dynamic error message based on the edit validation.

26.2.3 PA Reject Errors

Table 46: PA Reject Error Codes lists the types of error codes and their descriptions. These codes are used to populate the ErrorInfo object inside the error PA Reject notification. Refer to Section 11.2.3 Error Response to PA Request for more details.

Table 46: PA Reject Error Codes

Category	Error Code	Description
Requester	44	First and/or Last name is/are missing
Requester	35	Not a pilot participant State
Requester	51	NPI is missing or invalid
Requester	51	NPI does not match the Name of the Physician
Requester	51	Requester NPI is not on File
Requester	41	Provider is exempted from submitting this PA request
Requester	97	Provider address is missing or invalid
Requester	97	Provider city is missing or invalid
Requester	47	Requester state is missing or invalid
Requester	97	Provider zip is missing or invalid
Requester	15	Requested application data missing
Beneficiary	58	Date or Birth is missing or invalid
Beneficiary	44	First and/or Last name is/are blank
Beneficiary	66	Gender code is missing or invalid
Beneficiary	73	MBI number and name combination - invalid
Beneficiary	72	MBI number is missing or invalid
Beneficiary	95	Not eligible for service
Patient Event	AF	Diagnosis Code is missing or invalid
Patient Event	AF	Diagnosis code qualifier is missing or invalid
Facility	44	Name is missing
Facility	35	Not a pilot participant state
Facility	51	NPI does not match the name of the physician
Facility	51	NPI is missing or invalid
Facility	47	Provider state is missing or invalid
Facility	51	NPI is sent but not found
Facility	97	Provider address is missing or invalid
Facility	97	Provider city is missing or invalid

Category	Error Code	Description
Facility	97	Provider zip is missing or invalid
Ordering MD	44	First and/or Last name is missing
Ordering MD	35	Not a pilot participant state
Ordering MD	51	NPI does not match the name of the physician
Ordering MD	51	NPI is missing or invalid
Ordering MD	47	Provider state is missing or invalid
Ordering MD	51	NPI is sent but not found
Ordering MD	97	Provider address is missing or invalid
Ordering MD	97	Provider city is missing or invalid
Ordering MD	97	Provider zip is missing or invalid
Rendering MD/Supplier	44	First and/or Last name is missing
Rendering MD/Supplier	35	Not a pilot participant state
Rendering MD/Supplier	51	NPI does not match the name of the physician
Rendering MD/Supplier	51	NPI is missing or invalid
Rendering MD/Supplier	47	Provider state is missing or invalid
Rendering MD/Supplier	51	NPI is sent but not found
Rendering MD/Supplier	97	Provider address is missing or invalid
Rendering MD/Supplier	97	Provider city is missing or invalid
Rendering MD/Supplier	97	Provider zip is missing or invalid
Referring Provider	44	First and/or Last name is missing
Referring Provider	35	Not a pilot participant state
Referring Provider	51	NPI does not match the name of the physician
Referring Provider	51	NPI is missing or invalid
Referring Provider	47	Provider state is missing or invalid
Referring Provider	51	NPI is sent but not found
Referring Provider	97	Provider address is missing or invalid
Referring Provider	97	Provider city is missing or invalid
Referring Provider	97	Provider zip is missing or invalid

Category	Error Code	Description
Operating	44	First and/or Last name is missing
Operating	35	Not a pilot participant state
Operating	51	NPI does not match the name of the physician
Operating	51	NPI is missing or invalid
Operating	47	Provider state is missing or invalid
Operating	51	NPI is sent but not found
Operating	97	Provider address is missing or invalid
Operating	97	Provider city is missing or invalid
Operating	97	Provider zip is missing or invalid
Attending	44	First and/or Last name is missing
Attending	35	Not a pilot participant state
Attending	51	NPI does not match the name of the physician
Attending	51	NPI is missing or invalid
Attending	47	Provider state is missing or invalid
Attending	51	NPI is sent but not found
Attending	97	Provider address is missing or invalid
Attending	97	Provider city is missing or invalid
Attending	97	Provider zip is missing or invalid
Service	AG	Procedure code is missing
Service	AG	Procedure code qualifier is missing or invalid
Service	57	Proposed date/date range is missing or invalid
Service	AG	Procedure Code(s) is invalid
Service	15	Number of units is missing or invalid
Service	33	Place of service code is missing or invalid
Service	AG	Incorrect modifier for the procedure code
Service	57	Procedure code is repeated – same billing period (Future release)
Service	57	Date of service is invalid

27. PA Requests and Responses Automation with Shared Systems

27.1 Introduction

PA requests and responses are exchanged between the Providers and RCs via mail and fax as well as through the esMD system. esMD allows the exchange of PA information in electronic format as Accredited Standards Committee (ASC) X12N 278 transactions (requests/responses) along with the current acceptable format as XDR transactions. The corresponding medical documentation to the PA request is in XDR (PDF) format only.

27.1.1 Overview of the Automation Process

Currently, populating the PA screens in the Shared Systems is a manual process that is laborious and time consuming. The RCs receive the requests, manually enter the information, and respond with a written response or a response entered into RC Client. With the automation of PA requests/responses, esMD will intake the PA requests, automatically send the requests into the Shared System PA Screens and process the finalized PA requests sent from Shared Systems. This implementation will remove the manual data entry of X12N 278 PA request information into the PA screens by the RCs.

Refer to sections 27.2.1 Logical Workflow and 27.2.2 Application Workflow for detailed information on the automation processing of PA requests and responses with Shared System/Workloads.

27.1.2 Shared Systems

The automation of PA requests/responses will be implemented at different timelines by each of the Shared Systems (Multi-Carrier System (MCS), Viable Information Processing System (VIPS) Medicare System (VMS), and Fiscal Intermediary Shared System (FISS)).

In October 2016, release AR2016.10.0 implemented the changes in the esMD System to cover the initial rollout changes at MCS and Part B RCs.

27.1.2.1 PA Review Response

The X12N 278 Part B and XDR PA Review Response can be submitted using the Shared System PA Screens.

Assumptions

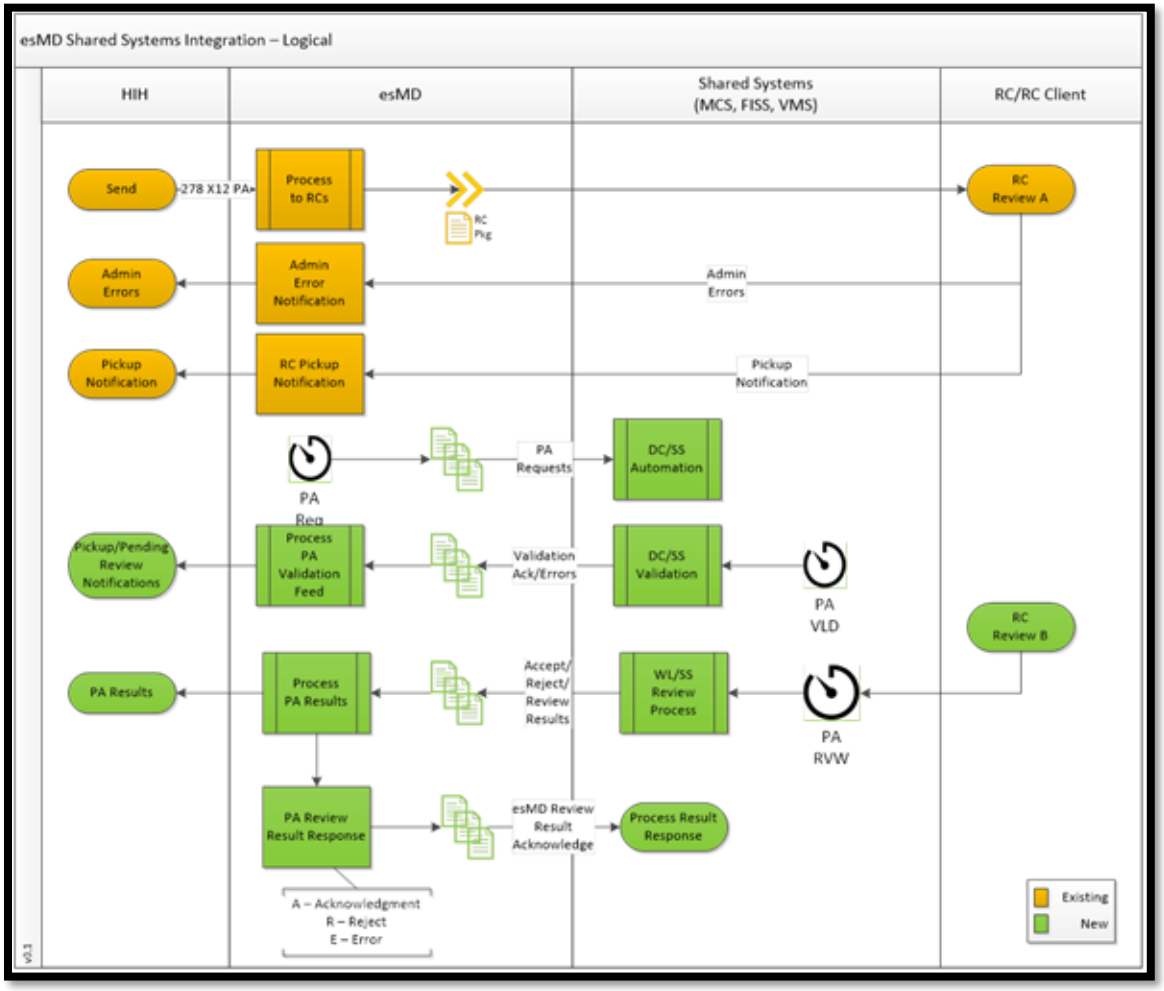
1. No User Interface feature will be available for eMDR and ICDT; only the API will be provided to support the ICDT, RRL, and PA/PCR functionality.

27.2 Automation of PA Requests/Responses – Application Workflow

27.2.1 Logical Workflow

Figure 97: esMD Shared System Integration - Logical provides an overview of the logical flow of PA Requests/Responses between esMD and Shared System (data center or workload).

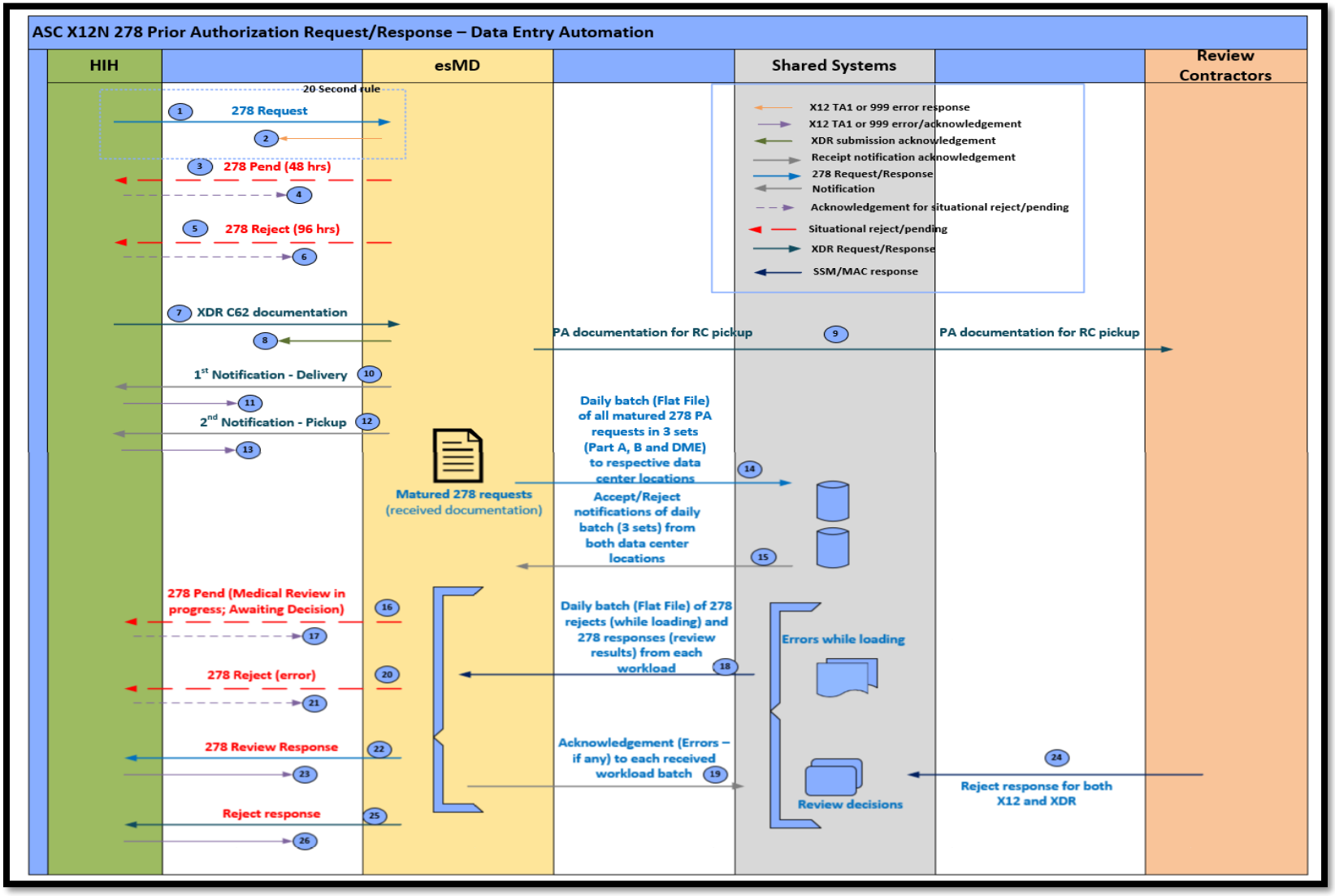
Figure 97: esMD Shared System Integration - Logical



27.2.2 Application Workflow

Figure 98: Information Flow – X12N 278 PA Request/Response Integration with Shared Systems provides an overview of the workflow of automation of X12N 278 PA Requests/Responses between esMD and Shared System/Workload.

Figure 98: Information Flow – X12N 278 PA Request/Response Integration with Shared Systems



28. Inbound/Outbound File Names and Data Directories

Table 47: Inbound/Outbound File Names and Data Directories lists all the files received by the RC and the corresponding data directories these files will reside in along with a brief description.

Note: GUID refers to esMD Transaction ID.

Table 47: Inbound/Outbound File Names and Data Directories

Data Directory	Program Type	Folder Names	XML File Name	Notes
Input	ADR Request (CTC-1)	E_L1_esMDTransactionID	E_L1_esMDTransactionID_metadata.xml	N/A
Error	ADR Validation error response	F_L1_esMDTransactionID	F_L1_esMDTransactionID_Validation_Error.xml	N/A
Notification	ADR HIH delivery notification	N_L1_esMDTransactionID_MMDDYY_HHMM	N_L1_esMDTransactionID_Delivery_Acknowledgement.xml	N/A
N/A	ADR Pickup Notification	P_L1_esMDTransactionID	P_L1_esMDTransactionID_Pickup.xml	N/A
Input	PWK Unsolicited documents (CTC-7)	E_L7_esMDTransactionID	E_L7_esMDTransactionID_metadata.xml	N/A
Error	PWK Unsolicited Validation error response	F_L7_esMDTransactionID	F_L7_esMDTransactionID_Validation_Error.xml	N/A
Notification	PWK Unsolicited HIH delivery notification	N_L7_esMDTransactionID_MMDDYY_HHMM	N_L7_esMDTransactionID_Delivery_Acknowledgement.xml	N/A
N/A	PWK Unsolicited Pickup Notification	P_L7_esMDTransactionID	P_L7_esMDTransactionIDID_Pickup.xml	N/A
Input	First level Appeals (CTC-9)	E_L9_esMDTransactionID	E_L9_esMDTransactionID_metadata.xml	N/A
Error	First Level Appeals Validation error response	F_L9_esMDTransactionID	F_L9_esMDTransactionID_Validation_Error.xml	N/A
Notification	First Level Appeals HIH Delivery Notification	N_L9_esMDTransactionID_MMDDYY_HHMM	N_L9_esMDTransactionID_Delivery_Acknowledgement.xml	N/A

Data Directory	Program Type	Folder Names	XML File Name	Notes
N/A	Pickup Notification	P_L9_esMDTransactionID	P_L9_esMDTransactionID_Pickup.xml	N/A
Input	Second Level Appeals (CTC 9.1)	E_L9_1_esMDTransactionID	E_L9_1_esMDTransactionID_metadata.xml	N/A
Error	Second Level Appeals Validation error response	F_L9_1_esMDTransactionID	F_L9_1_esMDTransactionID_Validation_Error.xml	N/A
notification	Second Level Appeals HIH delivery notification	N_L1_esMDTransactionID_MMDDYY_HHMM	N_L9_1_esMDTransactionID_Delivery_Acknowledgement.xml	N/A
N/A	Second Level Appeals Pickup Notification	P_L9_1_esMDTransactionID	P_L9_1_esMDTransactionID_Pickup.xml	N/A
input	ADMC (CTC 10)	E_L10_esMDTransactionID	E_L10_esMDTransactionID_metadata.xml	N/A
error	ADMC Validation error response	F_L10_esMDTransactionID	F_L10_esMDTransactionID_Validation_Error.xml	N/A
notification	ADMC HIH delivery notification	N_L10_esMDTransactionID_MMDDYY_HHMM	N_L10_esMDTransactionID_Delivery_Acknowledgement.xml	N/A
N/A	ADMC Pickup Notification	P_L10_esMDTransactionID	P_L10_esMDTransactionID_Pickup.xml	N/A
input	RAC Discussion Request (CTC 11)	E_L11_esMDTransactionID	E_L11_esMDTransactionID_metadata.xml	N/A
error	RAC Discussion Validation error response	F_L11_esMDTransactionID	F_L11_GUID_Validation_Error.xml	N/A
notification	RAC Discussion HIH delivery notification	N_L11_esMDTransactionID_MMDDYY_HHMM	N_L11_GUID_Delivery_Acknowledgement.xml	N/A
N/A	RAC Discussion Pickup Notification	P_L11_esMDTransactionID	P_L11_esMDTransactionID_Pickup.xml	N/A
input	Phone Discussion Request (CTC 11.1)	E_L11_1_GUID	E_L11_1_esMDTransactionID_metadata.xml	N/A
error	Phone Discussion Validation error response	F_L11_1_esMDTransactionIDesMDTransactionID	F_L11_1_esMDTransactionID_Validation_Error.xml	N/A
notification	Phone Discussion HIH delivery notification	N_L11_1_esMDTransactionID_MMDDYY_HHMM	N_L11_1_esMDTransactionID_Delivery_Acknowledgement.xml	N/A

Data Directory	Program Type	Folder Names	XML File Name	Notes
N/A	Phone Discussion Pickup Notification	P_L11_1_esMDTransactionID	P_L11_1_esMDTransactionID_Pickup.xml	N/A
input	Ambulance (CTC 8.1)	E_L8_1_esMDTransactionID	E_L8_1_esMDTransactionID_metadata.xml	N/A
error	Ambulance Validation error response	F_L8_1_esMDTransactionID	F_L8_1_esMDTransactionID_Validation_Error.xml	N/A
notification	Ambulance HIH delivery notification	N_L8_1_esMDTransactionID_MMDDYY_HHMM	N_L8_1_esMDTransactionID_Delivery_Acknowledgement.xml	N/A
N/A	Ambulance Pickup Notification	P_L8_1_esMDTransactionID	P_L8_1_esMDTransactionID_Pickup.xml	N/A
input	HHPCR (CTC 8.3)	E_L8_3_esMDTransactionID	E_L8_3_esMDTransactionID_metadata.xml	N/A
error	HHPCR Validation error response	F_L8_3_esMDTransactionID	F_L8_3_esMDTransactionID_Validation_Error.xml	N/A
notification	HHPCR HIH delivery notification	N_L8_3_esMDTransactionID_MMDDYY_HHMM	N_L8_3_esMDTransactionID_Delivery_Acknowledgement.xml	N/A
N/A	HHPCR Pickup Notification	P_L8_3_esMDTransactionID	P_L8_3_esMDTransactionID_Pickup.xml	N/A
input	DMEPOS (CTC 8.4)	E_L8_4_esMDTransactionID	E_L8_4_esMDTransactionID_metadata.xml	N/A
error	DMEPOS Validation error response	F_L8_4_esMDTransactionID	F_L8_4_esMDTransactionID_Validation_Error.xml	N/A
notification	DMEPOS HIH delivery notification	N_L8_4_esMDTransactionID_MMDDYY_HHMM	N_L8_4_esMDTransactionID_Delivery_Acknowledgement.xml	N/A
N/A	DMEPOS Pickup Notification	P_L8_4_esMDTransactionID	P_L8_4_esMDTransactionID_Pickup.xml	N/A
input	X12 XDR (CTC 12)	E_L12_esMDTransactionID	E_L12_esMDTransactionID_metadata.xml	N/A
error	Validation error response	F_L12_esMDTransactionID	F_L12_esMDTransactionID_Validation_Error.xml	N/A
notification	HIH delivery notification	N_L12_esMDTransactionID_MMDDYY_HHMM	N_L12_esMDTransactionID_Delivery_Acknowledgement.xml	N/A
N/A	Pickup Notification	P_L12_esMDTransactionID	P_L12_esMDTransactionID_Pickup.xml	N/A
input	Additional Documentation X12 XDR (CTC 13)	E_L13_esMDTransactionID	E_L13_esMDTransactionID_metadata.xml	N/A

Data Directory	Program Type	Folder Names	XML File Name	Notes
error	Validation error response	F_L13_esMDTransactionID	F_L13_esMDTransactionID_Validation_Error.xml	N/A
notification	HIH delivery notification	N_L3_esMDTransactionID_MMDDYY_HHMM	N_L13_esMDTransactionID_Delivery_Acknowledgement.xml	N/A
N/A	Pickup Notification	P_L13_esMDTransactionID	P_L13_esMDTransactionID_Pickup.xml	N/A
N/A	Admin Error Response	D_L1 esMDTransactionID	D_L1_ esMDTransactionID_ValidationError.xml	The Folder Name and XML File Name are the same for all the following Content Type Codes: CTC-1, CTC-7, CTC-8.1, , CTC-8.3, CTC-8.4, CTC-9, CTC-9.1, CTC-10, CTC-11, CTC-11.1, CTC-12, and CTC-13
ICDT/input	ICDT - Solicited Request	Q_Q<<PackageUniqueID>>	<<ReceiverRoutingId>>.T.L15_1.Q<<esMDTransactionID>>.<<SenderRoutingID>>.DMMddyy.THHmmssS	Q_Q<<PackageUniqueID>>_ICDTSolicitedRequest.xml
ICDT/input	ICDT - Solicited Response	R_R<<PackageUniqueID>>>>	<<ReceiverRoutingId>>.T.L15_2.R<<esMDTransactionID>>.<<SenderRoutingID>>.DMMddyy.THHmmssS	R_R<<PackageUniqueID>>ICDTSolicitedResponse.xml
ICDT/input	ICDT - Unsolicited Response	R_R<<PackageUniqueID>>	<<ReceiverRoutingId>>.T.L15_3.R<<esMDTransactionID>>.<<SenderRoutingID>>.DMMddyy.THHmmssS	R_R<<PackageUniqueID>>_ICDTUnSolicitedResponse.xml
Icdt\ntfn_ack	ICDT – Pickup Notifications/Acknowledgment	B_PackageUniqueID	N/A	B_<<PackageUniqueID>>_Pickup_Notification.xml B_<<PackageUniqueID>>_Delivery_Acknowledgment.xml
Icdt\error	ICDT - Admin Error Response	C_<<PackageUniqueID>>	N/A	C_<<PackageUniqueID>>_Admin_Notification.xml

Data Directory	Program Type	Folder Names	XML File Name	Notes
Icdt\error	ICDT Solicited Request - Validation errors from esMD	V_PackageUniqueID	N/A	V_<<Packageunique id>>_Validation_Error.xml
Icdt\error	ICDT Solicited Response - Validation errors from esMD	V_PackageUniqueID	V_GFT0007137629EC_Validation_Error.xml	V_<<Packageunique id>>_Validation_Error.xml
Icdt\error	ICDT Unsolicited Response - Validation errors from esMD	V_PackageUniqueID	V_VGEJ110822152126_Validation_Error.xml	V_<<Packageunique id>>_Validation_Error.xml
eMDRRegistration	eMDR Registration Batch File from esMD	E_esMDTransactionID	<<ReceiverRoutingId>>.T.L5.E<<esMDTransacstionID>>.<<SenderRoutingID>>.DMMddyy.THHmmssS	N/A
N/A	Pre-Pay eMDR letters (CTC 2.5)	U_UniqueId	U_UniqueID_eMDR_ProcessMetadata.xml	UniqueId-<<deliveryType>><<3 digit random number>><<MMddyyHHmmss>>
Acknowledgment	Pre-Pay eMDR letters Acknowledgment	A_L2_5_GUID_MMDDYY_HHMMSS	A_L2_5_GUID_Receipt_Acknowledgement.xml	N/A
error	Pre-Pay eMDR letters Validation error response	F_L2_5_GUID_MMDDYY_HHMMSS	F_L2_5_GUID_Validation_Error.xml	N/A
N/A	Post-Pay eMDR letters (CTC 2.6)	U_UniqueId	U_UniqueID_eMDR_ProcessMetadata.xml	UniqueId-<<deliveryType>><<3 digit random number>><<MMddyyHHmmss>>
Acknowledgment	Post-Pay eMDR letters Acknowledgment	A_L2_6_GUID_MMDDYY_HHMMSS	A_L2_6_esMDTransactionID_Receipt_Acknowledgement.xml	N/A
error	Post-Pay eMDR letters Validation error response	F_L2_6_GUID_MMDDYY_HHMMSS	F_L2_6_esMDTransactionID_Validation_Error.xml	N/A

Data Directory	Program Type	Folder Names	XML File Name	Notes
N/A	Post-Pay-Other eMDR letters (CTC 2.6)	W_UniqueId	W_UniqueID_eMDR_ProcessMetadata.xml	UniqueId-<<deliveryType>><<3 digit random number>><<MMddyyHHmmss>>
Acknowledgment	Post-Pay-Other eMDR letters Acknowledgment	A_L2_6_GUID_MMDDYY_HHMMSS	A_L2_6_GUID_Receipt_Acknowledgement.xml	N/A
N/A	Post-Pay-Other eMDR Structured File	W_UniqueId	W_WGUI185831202209_eMDRStructuredFile.xml	N/A
N/A	Post-Pay eMDR Structured File	U_UniqueId	U_UGUI185831202209_eMDRStructuredFile.xml	N/A
Input	Document Code Batch File from esMD	E_3CharRandomId	<<ReceiverRoutingId>>.T.L17.E<<esMDTransactionID>>.<<SenderRoutingID>>.DMMddyy.THHmmssS	N/A
input	HOPD (CTC 8.5)	E_L8_5_esMDTransactionID	E_L8_5_esMDTransactionID_metadata.xml	N/A
Input	IRF (CTC 8.6)	E_L8_6_esMDTransactionID	E_L8_6_esMDTransactionID_metadata.xml	N/A
error	HOPD Validation error response	F_L8_5_esMDTransactionID	F_L8_5_esMDTransactionID_Validation_Error.xml	N/A
Error	IRF Validation error response	F_L8_6_esMDTransactionID	F_L8_6_esMDTransactionID_Validation_Error.xml	N/A
notification	HOPD HIH delivery notification	N_L8_5_esMDTransactionID_MMDDYY_HHMM	N_L8_5_esMDTransactionID_Delivery_Acknowledgement.xml	N/A
notification	HOPD HIH delivery notification	N_L8_5_esMDTransactionID_MMDDYY_HHMM	N_L8_5_esMDTransactionID_Delivery_Acknowledgement.xml	N/A
N/A	HOPD Pickup Notification	P_L8_5_esMDTransactionID	P_L8_5_esMDTransactionID_Pickup.xml	N/A
N/A	IRF Pickup Notification	P_L8_6_esMDTransactionID	P_L8_6_esMDTransactionID_Pickup.xml	N/A
error	Letters Validation Error	data/error/LETTERS	LETTERS_F_esMDTransactionID_MMDDYY_HHMMSS.json	N/A
PDF File	eMDR Prepay	N/A	TESTLETTERID5678_20190405_REVIEW1.pdf	N/A

Data Directory	Program Type	Folder Names	XML File Name	Notes
notification	Letters HIH Delivery Notification Error	Data/notification/LETTERS	_N_esMDTransactionID_MMDDYY_HHMMSS.json	N/A

29. Contacts

Table 48: Support Points of Contact provides the contact information for the esMD Service Desk.

Table 48: Support Points of Contact

Contact	Phone	Email	Hours of Operation
CMS esMD Service Desk	(443) 832-1856	esMD_Support@cms.hhs.gov	Regular Business Hours: 8 a.m. to 8 p.m. Eastern Time (ET).

Appendix A: Description of Fields on RC Client Tabs

Table 49: Descriptions of Fields on Error Response to PA Request Tab lists the descriptions of the fields on the Review Decision Response to PA Request tab.

Table 49: Descriptions of Fields on Error Response to PA Request Tab

Name of Field	Description
Transaction ID	The esMD TransactionId format is as follows: • The length of TransactionId will be 15 alphanumeric characters. • TransactionId will consist of alphabetic (a-z, A-Z) and numeric (0-9) characters.
Reject Error Category	One or multiple Reject Error Category is selected for each Response; each Reject Error Category has number of Reject Error Codes associated with it. Required Element.
Reject Error Code	Under Each Reject Error Category, either one or multiple Reject Error Codes are selected. Required Element. Minimum 1 and maximum 9 reject error codes can be selected for each category.
Decision Indicator	Decision provided for the Error Response should be the following: • "R"- Decision Indicator. "R" is provided in the response when the Decision is "Rejected" for the request received.
Reason Code	5-character reason code is provided. Minimum of 1 and up to maximum of 25 reason codes can be provided. Required Element.
Request Level Unique Tracking Number (UTN)	UTN is provided for each response. Optional Element. Format of the unique tracking number is 14 Alpha Numeric Characters.

Table 50: Descriptions of Fields on Administrative Error Response to Inbound Submissions Tab lists the descriptions of the fields on the Administrative Error Response to Inbound Submissions tab.

Table 50: Descriptions of Fields on Administrative Error Response to Inbound Submissions Tab

Name of Field	Description
Transaction ID	Transaction Identifier of the request this response is being sent for. Required Element. It is a 15 Alphanumeric Value.
Error Situation	Error code/situation; can be one of the following: • Cannot Read Files/Corrupt Files • virus found • Submission sent to incorrect RC • Other • Incomplete File • Unsolicited Response • Documentation cannot be matched to a case/claim • Duplicate

Table 51: Descriptions of Fields on Advanced/Debugging Tab lists the descriptions of the fields on the Advanced/Debugging tab.

Table 51: Descriptions of Fields on Advanced/Debugging Tab

Name of Field	Description
User ID	IDM User ID Required Element for testing the connectivity to esMD Cloud environment.
Password	IDM password Required Element for testing the connectivity to esMD Cloud environment.

Appendix B: Reject Error Codes

For an up-to-date list of Reject Error Codes, please refer to the esMD Downloads section, using the link below:

http://www.cms.gov/Research-Statistics-Data-and-Systems/Computer-Data-and-Systems/ESMD/Information_for_Review-Contractors.html

Note: An up-to-date list of Reject Error Codes will be added to this web site by CMS.

Appendix C: Industry Codes

For an up-to-date list of Industry Codes, please refer to the esMD Downloads section, using the link below:

http://www.cms.gov/Research-Statistics-Data-and-Systems/Computer-Data-and-Systems/ESMD/Information_for_Review-Contractors.html

Note: An up-to-date list of Industry Codes will be added to this web site by CMS.

Appendix D: Content Type Codes

Table 52: Content Type Code Descriptions provides the descriptions of the Content Type Codes.

Table 52: Content Type Code Descriptions

Content Type Code	Description	Comment
1	Response to ADR	N/A
2.5	Pre-Pay eMDR letters	Pre-Pay eMDR letters from RC to esMD
2.6	Post-Pay/Post-Pay-Other eMDR letters	Post-Pay/Post-Pay-Other eMDR letters from RC to esMD
5	Service Registration	Service Registration
7	PWK Unsolicited documents	PWK Unsolicited documents
8.1	Non-Emergent Ambulance Transport	N/A
8.3	HHPCR	N/A
8.4	DMEPOS	N/A
8.5	HOPD	N/A
8.6	Inpatient Rehabilitation Facility (IRF)	N/A
9	First Level Appeal Requests	N/A
9.1	Second Level Appeal Requests	N/A
10	ADMC	N/A
11	RA Requests	N/A
11.1	DME Phone Discussion Requests	N/A
13	Supporting Documentation for the X12N 278 Request	N/A
15.1	ICDT Request	Supports requests for documentation from an RC to another RC.
15.2	ICDT Solicited Response	Supports responses from an RC for previously requested documentation to another RC.
15.3	ICDT Unsolicited Response	Supports an RC sending misdirected documentation to another RC.
17	Document Codes	Document Codes
20	Prior Authorization Decision Letter/Review Result Letter	Prior Authorization Decision Letter/Review Result Letter

Table 53: Content Type Codes and Business Types shows the list of Business Types associated with them.

Table 53: Content Type Codes and Business Types

Content Type Code	Business Type
1	Response message for additional documentation request
2.5	Pre-Pay eMDR
2.6	Post-Pay/Post-Pay-Other eMDR
5	Service Registration
7	PWK Unsolicited documents
8.1	Non-Emergent Ambulance Transport.
8.3	HHPCR
8.4	DMEPOS PA
8.5	HOPD
8.6	IRF
9	First Level Appeal
9.1	Second Level Appeal
10	ADMC
11	RA Requests
11.1	DME Phone Discussion Requests
13	XDR X12
15.1	ICDT Request
15.2	ICDT Solicited Response
15.3	ICDT Unsolicited Response
17	Document Code File
20	Prior Authorization Decision Letter/Review Result Letter

Appendix E: ADR Categorization Values

Table 54: ADR Categorization Value Descriptions provides the new ADR response metadata element values.

Table 54: ADR Categorization Value Descriptions

No	Description	Comment
1.10	MR (Medical Review)	Responses to Targeted, Probe & Educate (TPE), Pre-pay, Post-pay reviews.
1.11	Non-MR (Non- medical Review)	Provider to distinguish based on type of response.
1.12	PA – Responses	Responses for PA/PCR requests.

Appendix F: Acronyms

Table 55: Acronyms

Acronym	Literal Translation
ACN	Attachment Control Number
ADMC	Advance Determination of Medicare Coverage
ADR	Additional Documentation Request
API	Application Programming Interface
ASC	Accredited Standards Committee
AUTH	Authentication
CAQH	Council for Affordable Quality Healthcare
CERT	Comprehensive Error Rate Testing
CHIP	Children's Health Insurance Program
CMS	Centers for Medicare & Medicaid Services
CTC	Content Type Code
DC	Data Center
DCF	Document Code File
DMAC	DMEPOS Medicare Administrative Contractor
DME	Durable Medical Equipment
DMEPOS	Durable Medical Equipment, Prosthetics, Orthotics and Supplies
DTO	Data Transfer Object
EFT	Enterprise File Transfer
eMDR	Electronic Medical Documentation Request
esMD	Electronic Submission of Medical Documentation
FFS	Fee-For-Service
GUID	Globally Unique Identifier
HHPCR	Home Health Services Pre-Claim Review
HICN	Health Insurance Claim Number
HIH	Health Information Handler
HL7	Health Level Seven International
HOPD	Hospital Outpatient Department
ICD	Interface Control Document
ICDT	Inter Contractor Document Transfer
ID	Identifier
IDM	Identity Management.
IRF	Inpatient Rehabilitation Facility
JSON	JavaScript Object Notation
Kbps	Kilobits Per Second
LOB	Line of Business
MAC	Medicare Administrative Contractor
MB	Megabytes

Acronym	Literal Translation
MCS	Multi-Carrier System
MFT	Managed File Transfer
MIME	Multipurpose Internet Mail Extension
NPI	National Provider Identifier
NwHIN	Nationwide Health Information Network
OID	Object Identifier or Organizational Identifier
PA	Prior Authorization
PADL	Prior Authorization Decision Letter
PCR	Pre-Claim Review
PDF	Portable Document Format
PROD	Production
PWK	Paperwork
QIC	Qualified Independent Contractor
RAC	Recovery Audit Contractor
RC	Review Contractor
REST	Representational State Transfer
RRL	Review Results Letter
RSA	Rivest, Shamir & Adleman
SHA	Secure Hash Algorithm
SMRC	Supplemental Medical Review Contractor
SSH	Secure Shell
SSL	Secure Sockets Layer
TLS	Transport Layer Security
UAT	User Acceptance Test
UI	User Interface
UPIC	Unified Program Integrity Contractor
URL	Universal Resource Locator
UTN	Universal Tracking Number
XDR	Cross-Enterprise Document Reliable Interchange
XML	Extensible Markup Language

Appendix G: Glossary

Table 56: Glossary

Glossary	Description
Additional Documentation Request (ADR)	Official letters sent to Providers from CMS RCs requesting additional documentation that is needed to process claims.
Advanced Determination of Medical Coverage (ADMC)	A voluntary program that allows Suppliers and Beneficiaries to request prior approval of eligible items (e.g., wheelchairs) before delivery of the items to the beneficiary.
CONNECT	CONNECT implements a flexible, open-source gateway solution that enables healthcare entities - Federal agencies or private-sector health organizations or networks - to connect their existing health information systems to the eHealth Exchange. CONNECT is fully functional out-of-the-box, while at the same time configurable and flexible to allow organizations to customize it to meet their needs and those of their existing health information systems.
Electronic Submission of Medical Documentation (esMD)	A new mechanism for submitting medical documentation via a secure internet gateway connecting Providers to the Centers for Medicare & Medicaid Services (CMS). In its second phase, esMD will allow Medicare RCs to electronically submit claim related Additional Document Request (ADR) letters, and other use case requests, to Providers when their claims are selected for review.
Health Information Handler (HIH)	A Health Information Handler (HIH) is defined as an organization that oversees and governs the exchange of health-related claim reviewer information from Provider to CMS esMD Gateway according to nationally recognized standards.
Inter Contractor Document Transfer ICDT)	A new functionality that allows RCs to exchange files/documents from one RC to another RC through the esMD system.
Interface	A well-defined boundary where direct contact between two different environments, systems, etc., occurs, and where information is exchanged.
Power Mobility Device (PMD) Prior Authorization (PA)	The CMS implemented a Prior Authorization process for scooters and power wheelchairs for people with Fee-For-Service Medicare. The PMD PA category is part of the DMEPOS PA program, which helps to ensure that a beneficiary's medical condition warrants their medical equipment under existing coverage guidelines. Moreover, the program will assist in preserving a Medicare beneficiary's ability to receive quality products from accredited suppliers.
Security	The physical, technological, and administrative safeguards used to protect individually identifiable health information.
Simple Object Access Protocol (SOAP)	Simple Object Access Protocol is a message exchange format for web services.
Transaction	Event or process (such as an input message) initiated or invoked by a user or system, regarded as a single unit of work and requiring a record to be generated for processing in a database.

Appendix H: FAQs

- RC Client is inactive, but the screen says “Login Successful. RC Client is Active”
    - The blue successful message that is shown while logging in “Login Successful. RC Client is Active” only means that the login was successful and RC client is now active at the time. If there is any temporary internet disconnection, RC Client will display the message “RC Client is not Active. Please login” and will stop pulling and pushing the document. Please restart the RC Client.
- RC Client is not working properly, when using multiple instances
    - It’s not advised to use multiple copies of RC Client simultaneously. Only use one copy at time.

      Note: Running multiple instances of the Java RC Client for the same jurisdiction could result in errors while pulling the files.
- RC Client is unable to download the files, every file is erroring out.
    - RC Client needs folder permission to download the files. It needs folder read/write permission to download and copy the files. Please check with your IT team if there is any such issue.
- User can log into the CMS portal (https://home.idm.cms.gov/) but not into RC Client
    - Please make sure your KeyStore is created and updated with new or reset password.

Appendix I: Approvals

The undersigned acknowledge that they have reviewed the Review Contractor (RC) Client Java User Guide and Installation Handbook AR2025.04.0, Version 18.1, and agree with the information presented within this document. Changes to this Guide will be coordinated with, and approved by, the undersigned, or their designated representatives.

Signature:		Date:	
Print Name:	Brenda Barksdale		
Title:	Contracting Officer's Representative		
Role:	CMS Approving Authority		