Mississippi Medicaid Enterprise

MS File Transfer System

Interface Specifications

Change history

Rev #	Date	Author	Section	Nature of Change
2.0	10/4/2017	Mike Morgan	1.1	Updated details regarding interfaces.
3.0	09/01/2022	Alejandro Acosta	All	Create MS DDI documentation version.
3.2	03/27/2023	Alejandro Acosta	All	Content revised.
3.3	03/29/2023	Alejandro Acosta	6.3	Password expiration policy.

Preface

This document is intended for software vendors who wish to develop applications that interact with File Transfer System, Mississippi Medicaid's file delivery and retrieval system. It was created and is maintained by Gainwell Technologies for the purpose of uploading and downloading HIPAA-compliant transactions.

Table of contents

List of figures

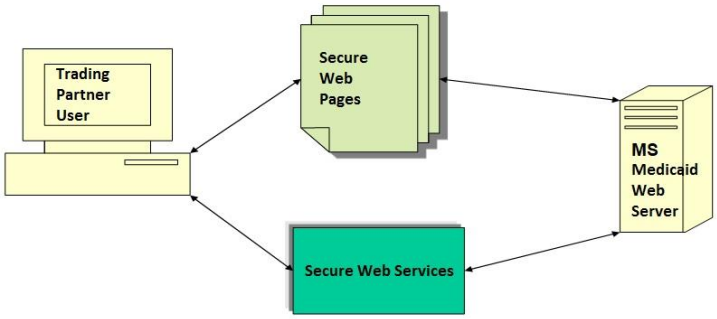
List of tables

1 Interface Standards

Mississippi Medicaid's File Transfer System interfaces support two of the most commonly used channels of communication, HTTP/S (web) and SFTP, giving clients a variety of interfaces to develop robust interchange solutions.

1.1 Web interface

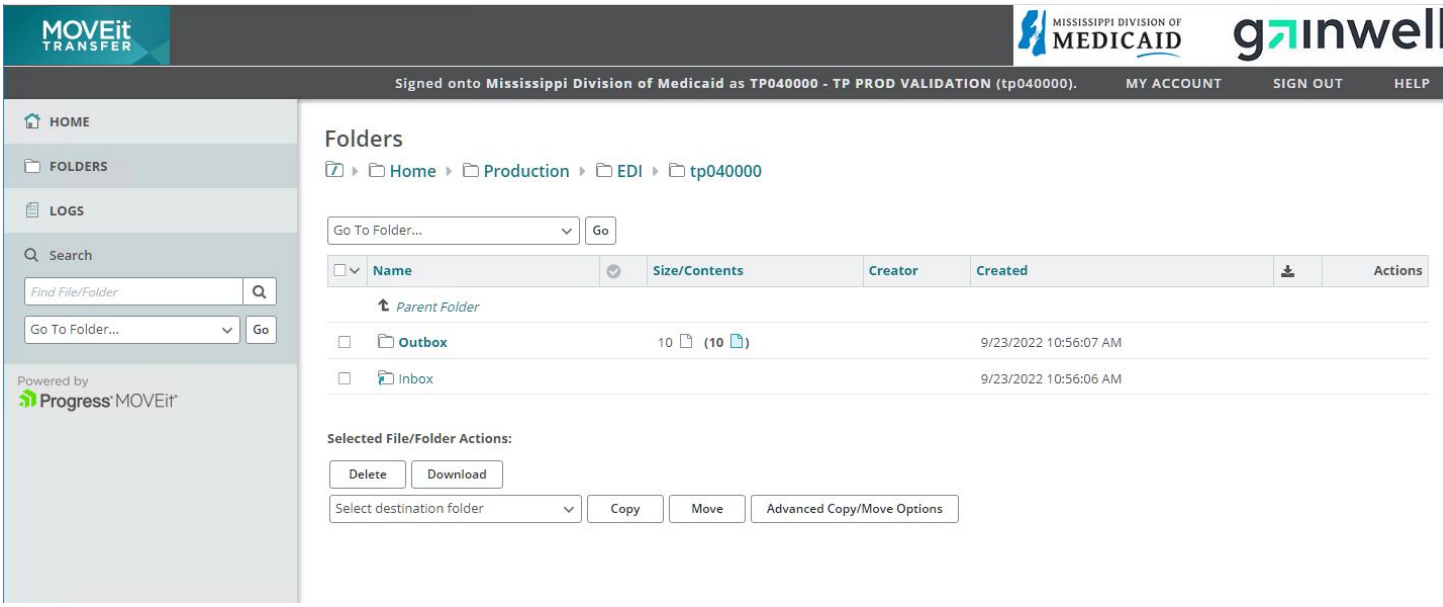
The web interface is comprised of two independent HTTP/S application interfaces; a website that supports only batch ASC X12/NCPDP transaction and a CAQH CORE Safe harbor interface capable of accepting both real-time and batch transactions. A diagram of the interaction flow is below.



1.1.1 Secure website

Mississippi Medicaid's secure website is located at https://fts.msxix.net/. The website is a batch-only interface integrated with file transfer wizards that allows users to upload and download files to and from the batch file repository. Once logged in, users can upload batch files to the secure server for processing. To upload batch files, navigate to the Inbox folder, and launch the upload/download wizard. To retrieve batch response files, navigate to the Outbox folder, and click the download link next to each file displayed. Complete user interface documentation can be found by clicking on the "Online Manual" link found in the menu on the left of the home screen. A screenshot of the website is provided below.

NOTE: Batch ASC X12/NCPDP files must be uploaded one at a time. Users cannot use the zip option within the upload/download wizard.

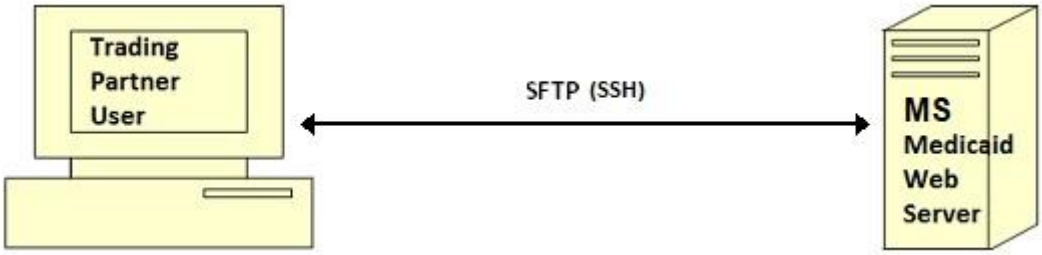


1.1.2 Secure CORE compliant web service

Mississippi Medicaid's CORE compliant web interface is designed around CORE ACA 1104 Phase I and II rules, found at http://www.caqh.org/pdf/CLEAN5010/PIv5010Complete.pdf and http://www.caqh.org/pdf/CLEAN5010/PIIv5010Complete.pdf, respectively. This service is capable of accepting two different HTTP/S transport envelope specifications: HTTP MIME-Multipart Form Data and SOAP+WSDL. Along with batch capability, this service performs 270 and 276 real-time transactions. All data is transmitted using the Secure Socket Layer (SSL), which encrypts data over the network. Complete documentation can be found in the "CORE Web Service Specifications" section.

2 SFTP interface

The SFTP interface supports batch file uploads and downloads only. Users may use SFTP (SSH) clients such Filezilla, Putty, and WS_FTP Pro to transfer files or develop software that will logon and transfer files programmatically using SSH protocol. Complete specifications can be found in the “SFTP Specifications” section of this document.



3 CORE Web Services

Mississippi Medicaid's CORE compliant web services are built around ACA 1104 Phase I and II rules which can be found at https://www.caqh.org/pdf/CLEAN5010/PIv5010Complete.pdf and https://www.caqh.org/pdf/CLEAN5010/PIIv5010Complete.pdf. These services also include:

- Acceptance of all Mississippi Medicaid approved batch transactions, including NCPDP D.0 and MS census (MSCN)
- Retrieval of NCPDP D.0 and Mississippi Medicaid specific MSCN supplemental reject results

The WSDL for the SOAP+WSDL service can be obtained from CORE at https://www.caqh.org/SOAP/WSDL/CORERule2.2.0.wsdl. Data contracts can be found at https://www.caqh.org/SOAP/WSDL/CORERule2.2.0.xsd. Security policies, application endpoints, and Mississippi Medicaid-specific method definitions can be found on the Gainwell Technologies server at https://core.msxix.net/Soap/COREservice.svc?wsdl.

3.1 HTTP/S envelope standards

Mississippi Medicaid's CORE compliant web services are capable of accepting all Mississippi-approved transactions from the following two HTTP/S envelope standards: MIME-Multipart form data and SOAP+WSDL. Each envelope should conform to the CORE requirements listed below:

MIME-Multipart Form Data (IETF RFC 2388)

- HTTP/S version 1.1
- MIME version 1.0
- CORE envelope version 2.2.0

SOAP+WSDL

- HTTP/S version 1.1
- SOAP version 1.2
- WSDL version 1.1
- Web Service-Security 1.1
- CORE envelope version 2.2.0

3.2 Functionality at a glance

HTTP/S supports a request-response message pattern, meaning that the sender must submit a message and wait for a response from the recipient. This usage pattern applies to batch and real-time ASC X12/NCPDP transactions. The response message varies based on whether the sender's message was a real-time request, batch submission, or batch response retrieval request.

A list of the service functions can be seen below in Table 16. HTTP MIME-Multipart form data requests use the specified SOAP+WSDL methods. The functionality and CORE data envelope is inferred from the combination of processing mode and payload type. Request/response messages are described later in this guide.

Table 1: SOAP+WSDL web methods

Web Method	CORE	Supported
BatchSubmitTransaction	Yes	Yes
BatchSubmitAckRetrievalTransaction	Yes	Yes
BatchResultsRetrievalTransaction	Yes	Yes
BatchResultsAckSubmitTransaction	Yes	Yes
RealTimeTransaction	Yes	Yes
GenericBatchSubmissionTransaction	Yes	No
GenericBatchRetrievalTransaction	Yes	No
GenericBatchReceiptConfirmationTransaction	Yes	No

3.3 Production vs. model environments

There are 2 separate endpoints for production and test, which system you submit transactions to will determine how the will be processed.

WSDL can be obtained from the following URLs

- PROD: https://core.msxix.net/soap/coreservice.svc?wsdl
- PROD NCPDP: https://core-ncpdo.msxix.net/soap/coreservice.svc?wsdl
- TEST: https://core-tpi.msxix.net/soap/coreservice.svc?wsdl

When submitting transactions the endpoints are as follows

- PROD-BATCH : https://core.msxix.net/soap/coreservice.svc/batch
- PROD-REALTIME: https://core.msxix.net/soap/coreservice.svc/realtime
- TEST-BATCH: https://core-tpi.msxix.net/soap/coreservice.svc/batch
- TEST-REALTIME: https://core-tpi.msxix.net/soap/coreservice.svc/realtime

NOTE: From this point forward all connections will be in terms of production connection.

3.4 Realtime vs. batch processing mode

RealTime requests should contain only a single inquiry, i.e., only one eligibility inquiry per single information source for a single patient. RealTime responses for the receipt of an ASC X12/NCPDP 270 or 276 will be returned within 20 seconds. The response will be an error response or the corresponding ASC X12/NCPDP response (e.g., ASC X12/NCPDP v5010 TA1, v5010 999, or v5010 271, if the request submitted was a v5010 270). All RealTime ASC X12/NCPDP transaction data submitted or returned via SOAP 1.2 messaging standards should be sent in-line (within the SOAP message XML) as plain text wrapped within an XML CDATA tag. When using MIME Multi-part form data, the transaction data should be sent in-line (within the form content) as plain text; however, it should not be wrapped within an XML CDATA tag. Batch requests can be sent in the same format as RealTime requests but are not limited to single inquiries.

Responses to batch requests will differ due to ASC X12/NCPDP acknowledgment and transactional response time. An HTTP/S response message containing a CORE data envelope that indicates whether the request was accepted for processing or rejected will be returned within 60 seconds. Currently, Mississippi Medicaid only supports negative batch acknowledgements (a 999 or TA1 response). These

responses will be available for download within one (1) hour of the submission of an ASC X12/NCPDP 270 or 276 batch transaction. An ASC X12/NCPDP transactional response for the receipt of a batch v5010 270 or v5010 276 request submitted before 9:00 pm Eastern Time on any business day will be available for download no later than 7:00 am Eastern Time the following business day (seven (7) days a week, where a business day consists of a 24 hour period commencing at 12:00 am (Midnight or 00:00 hours) through 11:59 pm (23:59 hours) of the same day). Although not mandated by CORE operating rules, Mississippi Medicaid will have ASC X12/NCPDP acknowledgements (if any to report) and transactional ASC X12/NCPDP responses for other approved ASC X12/NCPDP batch transactions (837I, 837P, 837D) available for download within the same time frame.

All batch transactions sent and returned via SOAP 1.2 messaging should incorporate the transmission optimization mechanism, MTOM. MTOM is the reorganization of a SOAP 1.2 message into a MIME-Multipart SOAP message which allows the attachment of large payloads of data. ASC X12/NCPDP data should be sent as an MTOM MIME attachment comprised of a base64 encoded byte array. MTOM standards can be found at http://www.w3.org/TR/soap12-mtom/. When MIME Multi-part form data messaging is used, the payload should be transmitted in-line (within the form data content) as plain text.

3.5 Connecting to the Server

Users can connect to the web service using a network connection that provides access to the public internet. The envelope standards used determine the application endpoint URL that will be used to connect to and interact with Mississippi Medicaid's CORE services. Each interface listed below uses TLS 1.1 and above.

Service requests using SOAP 1.2 have a second determination for application end point. RealTime go to: https://core.msxix.net/soap/COREservice.svc/realtime. For NCPDP RealTime go to https://core-ncpdp.msxix.net/soap/COREservice.svc/realtime.

3.6 RealTime Processing

RealTime transactions may use either HTTP/S envelope standard covered in the previous sections. When using SOAP 1.2 messaging, the web method (seen in the "Functionality at Glance" section) will determine the appropriate CORE data envelope. However, when using MIME Multipart form data, the CORE data and intended functionality are inferred using the combination of processing mode and payload type. All RealTime transactions must have a processing mode of *RealTime* and one of the acceptable RealTime payload types covered in Appendix B. An HTTP/S response containing a RealTime CORE data envelope with an appropriate payload transactional response and an error code or description of envelope processing will be returned within 20 seconds. Consult the "Error Handling" section for more information regarding error situations. Descriptions of the RealTime CORE request and response data envelopes are listed in Tables 17 and 18.

3.6.1 RealTimeRequestMessage

RealTime transactions with the server should use a *RealTimeRequestMessage* comprised of an HTTP/S envelope and the COREEnvelopeRealTimeRequest envelope as listed in Table 17. Only one ASC X12/NCPDP envelope (ISA\IEA) containing a single inquiry may be submitted at a time and the total length in bytes of the entire message, including the HTTP/S envelope, must not exceed one (1) megabyte. If the message size exceeds this maximum, the entire request will be rejected. See Appendix B for a sample program.

Table 2: COREEnvelopeRealTimeRequest description

Element	Description	Required
UserName	The submitter's Trading Partner ID. When using SOAP 1.2 messaging, the value is sent in the Username token of the security element within the SOAP header. Type: String Sample: TP000001	Y
Password	The submitter's password. When using SOAP 1.2 messaging, this value is in the Password token of the security element in the SOAP header. Password requirements can be found the "Security standards and Practices" section of this guide. Type: String Sample: 1qaz@WSX	Y
PayloadType	The type of data being sent in the payload. A list of acceptable payload types can be found in Appendix A. Type: String Sample: X12_270_Request_005010X279A1	Y
ProcessingMode	The mode in which the request should be processed. A list of acceptable PayloadTypes can be found in Appendix A. Type: String Sample: RealTime	Y
PayloadID	A unique identifier within the domain sending the request. A PayloadID may not be used more than once for a given ProcessingMode and SenderID combination. Type: UUID Sample: f81d4fae-7dec-11d0-a765-00a0c91e6bf6	Y
TimeStamp	The date time, in UTC format, that the request was sent. Type: DateTime Sample: 2022-08-30T10:20:34Z	Y
SenderID	The ID of the sender. For all requests, this must match the UserName. Type: String Sample: TP000001	Y
ReceiverID	The ID of receiver. For all requests, this value must be '77032'. Type: String Sample: 77032	Y

Element	Description	Required
CORERuleVersion	The CORE envelope version being submitted. Mississippi Medicaid only supports CORERuleVersion 2.2.0 Type: String Sample: 2.2.0	Y
Payload	The ASC X12/NCPDP transactional data being sent for processing. Must be sent as plain text and when using SOAP 1.2 messaging, must be wrapped in a XML CDATA tag. Type: String Sample: ISA*00* *00* *ZZ*TP000001…IEA*1*000000031~	Y

3.6.2 Sample XML and XML schema

Figure 1: Sample SOAP 1.2 RealTimeRequestMessage

```
POST /Soap/COREService.svc/realtime HTTP/1.1
Host: core.msxix.net
Content-Type: application/soap+xml;charset=UTF-8;action="RealTimeTransaction"
Content-Length: 1385


<soap:Envelope xmlns:cor="HTTP/S://www.caqh.org/SOAP/WSDL/CORERule2.2.0.xsd"
                              xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
 <soap:Header>
  <wsse:Security soap:mustUnderstand="true" xmlns:wsse="HTTP/S://docs.oasis-
open.org/wss/2004/01/
            oasis-200401-wss-wssecurity-secext-1.0.xsd">
   <wsse:UsernameToken wsu:Id="UsernameToken-1" xmlns:wsu="HTTP/S://docs.oasis-open.org/wss/
              2004/01/oasis-00401-wss-wssecurity-utility-1.0.xsd">
     <wsse:Username>TP000001</wsse:Username>
     <wsse:Password Type="HTTP/S://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-
              token-profile-1.0#PasswordText">Zaq1XSW@</wsse:Password>
     <wsu:Created>2012-09-07T16:44:37.616Z</wsu:Created>
    </wsse:UsernameToken>
   </wsse:Security>
 </soap:Header>
  <soap:Body>
   <cor:COREEnvelopeRealTimeRequest>
    <PayloadType>X12_270_Request_005010X279A1</PayloadType>
    <ProcessingMode>RealTime</ProcessingMode>
```

```
    <PayloadID>f81d4fae-7dec-11d0-a765-00a0c91e6bf6</PayloadID>
    <TimeStamp>2022-08-30T10:20:34Z</TimeStamp>
    <SenderID>TP000001</SenderID>
    <ReceiverID>secure.mediciad.state.ar.us</ReceiverID>
    <CORERuleVersion>2.2.0</CORERuleVersion>
    <Payload>
      <![CDATA[ISA*00*         *00*         *ZZ*TP000001…IEA*1*000000031~]]>
    </Payload>
   </cor:COREEnvelopeRealTimeRequest>
 </soap:Body>
</soap:Envelope>
```

Figure 2: Sample MIME Multi-part form data RealTimeRequestMessage

```
POST /mime/COREservice.aspx HTTP/1.1
Host: core.msxix.net
Content-Length: 1385
Content-Type: multipart/form-data; boundary=MIME_BOUNDRY

--MIME_BOUNDRY
Content-Disposition: form-data; name="PayloadType"

X12_270_Request_005010X279A1
--MIME_BOUNDRY
Content-Disposition: form-data; name="ProcessingMode"

RealTime
--MIME_BOUNDRY
Content-Disposition: form-data; name="PayloadID"

f81d4fae-7dec-11d0-a765-00a0c91e6bf6
--MIME_BOUNDRY
Content-Disposition: form-data; name="TimeStamp"

2022-08-30T10:20:34Z
--MIME_BOUNDRY
```

```
Content-Disposition: form-data; name="UserName"

TP000001
--MIME_BOUNDRY
Content-Disposition: form-data; name="SenderID"

TP000001
--MIME_BOUNDRY
Content-Disposition: form-data; name="ReceiverID"

77032
--MIME_BOUNDRY
Content-Disposition: form-data; name="CORERuleVersion"

2.2.0
--MIME_BOUNDRY
Content-Disposition: form-data; name="Payload"

ISA*00*        *00*        *ZZ*TP000001…IEA*1*000000031~
--MIME_BOUNDRY--
```

3.6.3 RealTimeResponseMessage

Table 3: COREEnvelopeRealTimeResponse description

Element	Description	Require
PayloadType	An appropriate response payload type. A list of possible response payload types can be found in Appedix B. Type: String Sample: X12_271_Response_005010X279A1	Y
ProcessingMode	The processing mode submitted in the request. Type: String Sample: RealTime	Y
PayloadID	The payload ID submitted in the request. Type: GUID Sample: f81d4fae-7dec-11d0-a765-00a0c91e6bf6	Y

TimeStamp	The date and time, in UTC format, that the response was sent. Type: DateTime Sample: 2022-08-30T10:20:34Z	Y
SenderID	The ID of the sender. For all responses, this value must be '77032'. Type: String Sample: 77032	Y
ReceiverID	The ID of receiver. For all responses, this must match the Trading Partner ID that submitted the request. Type: String Sample: TP000001	Y
CORERuleVersion	The CORE envelope version being submitted. Mississippi Medicaid only supports CORERuleVersion 2.2.0 Type: String Sample: 2.2.0	Y
ErrorCode	The code of the error that occurred during message processing, not including ASC X12/NCPDP transactional errors. This field will have a value of 'Success' when a successful transmission occurs. A list of possible error codes may be found in "Error Handling" section of this guide. Type: String Sample: Unauthorized	Y
ErrorMessage	A friendly description of the error code. This field will be empty when a successful transmission occurs. Type: String Sample: Invalid username/password or not allowed to sign on from this location	Y
Payload	The ASC X12/NCPDP transactional response data. Returned in plain text and when using SOAP 1.2 messaging, must be wrapped in an XML CDATA tag. Type: String Sample: ISA*00*W2540582 *00* *30*…IEA*1*000000031~	Y

3.6.4 Sample XML and XML schema

Figure 3: Sample SOAP RealTimeResponseMessage

```
HTTP/1.1 200 OK
```

```
X-Powered-By: ASP.NET
X-AspNet-Version: 4.0.30319
Content-Length: 1155
Content-Type: application/soap+xml; charset=utf-8


<s:Envelope xmlns:s="HTTP/S://www.w3.org/2003/05/soap-envelope"  xmlns:u="HTTP/S://docs.oasis-open.org/
     wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
 <s:Header>
  <o:Security s:mustUnderstand="1" xmlns:o="HTTP/S://docs.oasis-open.org/wss/2004
       01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
   <u:Timestamp u:Id="_0">
    <u:Created>2012-09-07T16:44:43.898Z</u:Created>
    <u:Expires>2012-09-07T16:49:43.898Z</u:Expires>
   </u:Timestamp>
  </o:Security>
 </s:Header>
 <s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <COREEnvelopeRealTimeResponse xmlns="HTTP/S://www.caqh.org/SOAP/WSDL/CORERule2.2.0.xsd">
   <PayloadType xmlns="">X12_271_Response_005010X279A1</PayloadType>
   <ProcessingMode xmlns="">RealTime</ProcessingMode>
   <PayloadID xmlns="">f81d4fae-7dec-11d0-a765-00a0c91e6bf6</PayloadID>
   <TimeStamp xmlns="">2012-09-07T16:44:43Z</TimeStamp>
   <SenderID xmlns="">77032</SenderID>
   <ReceiverID xmlns="">TP000001</ReceiverID>
   <CORERuleVersion>2.2.0</CORERuleVersion>
   <ErrorCode xmlns="">Success</ErrorCode>
   <ErrorMessage xmlns=""></ErrorMessage>
   <Payload>
    <![CDATA[ISA*00*W2540582  *00*          *30*77032…IEA*1*000000031~]]>
   </Payload>
  </COREEnvelopeRealTimeResponse>
 </s:Body>
</s:Envelope>
```

NOTE: Some frameworks, such as .Net WCF, will unwrap the payload from the CDATA tag by default.

Figure 4: Sample MIME Multi-part form data RealTimeResponseMessage

```
HTTP/1.1 200 OK
Content-Length: 2408
Content-Type: multipart/form-data; boundary=MISS_MIMEBOUNDRY

--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="PayloadType"

X12_271_Response_005010X279A1
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="ProcessingMode"

RealTime
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="PayloadID"

f81d4fae-7dec-11d0-a765-00a0c91e6bf6
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="TimeStamp"

2022-08-30T10:20:34Z
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="SenderID"

77032
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="ReceiverID"

TP000001
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="CORERuleVersion"

2.2.0
--MISS_MIMEBOUNDRY
```

```
Content-Disposition: form-data; name="ErrorCode"

Success
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="ErrorDescription"

--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="Payload"

ISA*00*W2540582  *00*         *30*77032      *ZZ*TP000001…IEA*1*000000031~
--MISS_MIMEBOUNDRY--
```

3.7 Batch processing

Submission of batch transactions as a batch CORE data envelope should use either HTTP/S envelope standard covered. When using SOAP 1.2 messaging, the web methods determine the CORE data envelope that can be used to submit requests. When using the MIME Multipart form data standard, the CORE data envelope and intended functionality are inferred from the combination of processing mode and payload type. All batch requests must have a processing mode of *Batch* and one of the acceptable batch payload types covered in Appedix B. An HTTP/S response containing a batch CORE data envelope with an appropriate payload and an error code or description of processing will be returned within 60 seconds. Consult the “Error Handling” section for information regarding error situations. Descriptions of available batch CORE request/response data envelopes are listed in Table 19 and 20.

3.7.1 BatchSubmissionMessage

Upload of batch transaction files to the Gainwell Technologies batch repository should use a *BatchSubmissionMessage* comprised of the HTTP/S envelope and the *COREEnvelopeBatchSubmission* envelope described below. More than one batch transaction (i.e. ISA/IEA, if uploading ASCX12/NCPDP file) may be uploaded at a time by separating each of the files with the ASCII file separator character FS (0x1C). During the upload process, if any files fail to upload, those files (ISA/IEA envelopes) shall be returned in the payload of the response along with a PartialUpload error code. The user can retry the upload using a new PayloadID. The total length in bytes of an entire message, including the HTTP/S envelope, cannot exceed four (4) megabytes. If a message exceeds this maximum, the entire request will be rejected. See Appendix B for a sample program.

Table 4: COREEnvelopeBatchSubmission description

Element	Description	Required
UserName	The submitter’s Trading Partner ID. When using SOAP 1.2 messaging, the value is in the Username token of the security element within the SOAP header. Type: String Sample: TP000001	Y

Element	Description	Required
Password	The submitter's password. When using SOAP 1.2 messaging, this value is in the Password token of the security element in the SOAP header. Password requirements can be found in the "Security Standards and Practices" section of this document. Type: String Sample: 1qaz@WSX	Y
PayloadType	The type of data being sent in the payload. A list of acceptable payload types can be found in Appedix A. Type: String Sample: X12_270_Request_005010X279A1	Y
ProcessingMode	The mode in which the request should be processed. A list of acceptable processing modes can be fouind in Appedix B. Type: String Sample: Batch	Y
PayloadID	A unique identifier within the domain sending the request. A PayloadID may not be used more than once for a given ProcessingMode and SenderID combination. Type: UUID Sample: f81d4fae-7dec-11d0-a765-00a0c91e6bf6	Y
PayloadLength	The length of data being sent in the Payload. Type: int Sample: 1024	
TimeStamp	The date and time, in UTC format, that the request was sent. Type: DateTime Sample: 2022-08-30T10:20:34Z	Y
SenderID	The ID of the sender. For all requests, this must match the UserName. Type: String Sample: TP000001	Y
ReceiverID	The ID of receiver. For all requests, this value should be '77032'. Type: String Sample: 77032	Y
CORERulesVersion	The CORE envelope version being submitted. Mississippi Medicaid only supports CORERuleVersion 2.2.0.	Y

Element	Description	Required
	Type: String Sample: 2.2.0	
Checksum	A SHA-1 hash computed on the Payload contents. Type: String Sample: 6A3FE55946	Y
Payload	The transaction data being sent for processing. When using SOAP 1.2 messaging payloads, data must be sent as a base64 encoded byte array in an MTOM attachement. If using MIME Multi-part form data payloads, data must be sent as plain text and in-line. Type: String Sample: ISA*00* *00* *ZZ*TP000001…IEA*1*000000031~	Y

3.7.2 Sample XML and XML schema

Figure 5: Sample SOAP 1.2 MTOM BatchSubmissionMessage

```
POST /Soap/COREService.svc/batch HTTP/1.1
Content-Type: multipart/related;type="application/xop+xml";start="<rootpart@soapui.org>";
                start-info="application/soap+xml; action=\" BatchSubmitTransaction \"";
                boundary="----=_Part_0_5707285.1347309365508"
Host: core.msxix.net
MIME-Version: 1.0
Content-Length: 2781

------=_Part_0_5707285.1347309365508
Content-Type: application/xop+xml; charset=UTF-8; type="application/soap+xml; action=\"
BatchSubmitTransaction \""
Content-Transfer-Encoding: 8bit
Content-ID: <rootpart@soapui.org>

<soap:Envelope xmlns:cor="http://www.caqh.org/SOAP/WSDL/CORERule2.2.0.xsd"
xmlns:soap="http://www.w3.org/
        2003/05/soap-envelope">
  <soap:Header>
    <wsse:Security soap:mustUnderstand="true" xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/
            oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <wsse:UsernameToken wsu:Id="UsernameToken-1" xmlns:wsu="http://docs.oasis-
```

```
open.org/wss/2004/
                01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
          <wsse:Username>TP000001</wsse:Username>
          <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401 wss-username-
                token-profile-1.0#PasswordText">Zaq1XSW@</wsse:Password>
          <wsu:Created>2012-09-10T20:36:05.414Z</wsu:Created>
       </wsse:UsernameToken>
     </wsse:Security>
  </soap:Header>
  <soap:Body>
     <cor:COREEnvelopeBatchSubmission>
       <PayloadType>X12_270_Request_005010X279A1</PayloadType>
       <ProcessingMode>Batch</ProcessingMode>
       <PayloadID>f81d4fae-7dec-11d0-a765-00a0c91e6bf6</PayloadID>
       <PayloadLength>583</PayloadLength>
       <TimeStamp>2012-07-20T20:26:16.55Z</TimeStamp>
       <SenderID>TP000001</SenderID>
       <ReceiverID>77032</ReceiverID>
       <CORERuleVersion>2.2.0</CORERuleVersion>
       <CheckSum>b207582796442d81514f11d395b3d1f441485810</CheckSum>
       <Payload>
         <inc:Include href=cid:http://www.soapui.org/543267678132204
xmlns:inc="http://www.w3.org/2004/08/xop/include"/>
        </Payload>
     </cor:COREEnvelopeBatchSubmission>
  </soap:Body>
</soap:Envelope>

------=_Part_0_5707285.1347309365508
Content-Type: application/octet-stream
Content-Transfer-Encoding: binary
Content-ID: <http://www.soapui.org/543267678132204>

[Submitted data here]
------=_Part_0_5707285.1347309365508--
```

NOTE: The actual payload is located in the MIME-attachment referenced by the xop include element.

Figure 6: Sample MIME Multi-part form data BatchSubmissionMessage

```
POST /mime/COREservice.aspx HTTP/1.1
Host: core.msxix.net
Content-Length: 1385
Content-Type: multipart/form-data; boundary=MIME_BOUNDRY

--MIME_BOUNDRY
Content-Disposition: form-data; name="PayloadType"

X12_270_Request_005010X279A1
--MIME_BOUNDRY
Content-Disposition: form-data; name="ProcessingMode"

Batch
--MIME_BOUNDRY
Content-Disposition: form-data; name="PayloadID"

f81d4fae-7dec-11d0-a765-00a0c91e6bf6
--MIME_BOUNDRY
Content-Disposition: form-data; name="PayloadLength"

583
--MIME_BOUNDRY
Content-Disposition: form-data; name="TimeStamp"

20-08-30T10:20:34Z
--MIME_BOUNDRY
Content-Disposition: form-data; name="UserName"

TP000001
--MIME_BOUNDRY
Content-Disposition: form-data; name="SenderID"

TP000001
--MIME_BOUNDRY
```

```
Content-Disposition: form-data; name="ReceiverID"

77032
--MIME_BOUNDRY
Content-Disposition: form-data; name="CORERuleVersion"

2.2.0
--MIME_BOUNDRY
Content-Disposition: form-data; name="Checksum"

b207582796442d81514f11d395b3d1f441485810
--MIME_BOUNDRY
Content-Disposition: form-data; name="Payload"

ISA*00*         *00*         *ZZ*TP000001…IEA*1*000000031~
--MIME_BOUNDRY--
```

NOTE: The payload is plain text and in-line.

3.7.3 BatchSubmissionResponseMessage

Table 5: COREEnvelopeBatchSubmissionResponse description

Element	Description	Required
PayloadType	An appropriate response payload type. A list of possible response payload types can be found in Appedix B. Type: String Sample: X12_270_Request_005010X279A1	Y
ProcessingMode	The processing mode in which the request was submitted. Type: String Sample: Batch	Y
PayloadID	The PayloadID submitted in the request. Type: UUID Sample: f81d4fae-7dec-11d0-a765-00a0c91e6bf6	Y
TimeStamp	The date and time, in UTC format, that the response was sent. Type: DateTime Sample: 20-08-30T10:20:34Z	Y

SenderID	The ID of the sender. For all responses, this will be ‘77032’. Type: String Sample: 77032	Y
ReceiverID	The ID of receiver. For all responses, this must match the Trading Partner ID that submitted the request. Type: String Sample: TP000001	Y
CORERulesVersion	The CORE envelope version being submitted. Mississippi Medicaid only supports CORERuleVersion 2.2.0. Type: String Sample: 2.2.0	Y
ErrorCode	The code of the error that occurred during message processing, not including ASC X12/NCPDP transactional errors. This field will have a value of ‘Success’ when a successful transmission occurs. A list of possible error codes may be found in “Error Handling” section of this guide. Type: String Sample: Unauthorized	Y
ErrorMessage	A friendly description of the error code. This field will be empty when a successful transmission occurs. Type: String Sample: Invalid username/password or not allowed to sign on from this location	Y

3.7.4 Sample XML and XML schema

Figure 7: Sample SOAP 1.2 MTOM BatchSubmissionResponseMessage

```
HTTP/1.1 202 Accepted
Content-Length: 1418
Content-Type: multipart/related; type="application/xop+xml";start="<http://tempuri.org/0>";
        boundary="uuid:96ec3f73-ba4d-4d60-8c09-3645a3cb0b88+id=1";start-
info="application/soap+xml"
Server: Microsoft-IIS/7.5
MIME-Version: 1.0
X-Powered-By: ASP.NET
Date: Wed, 26 Sep 2012 19:38:38 GMT


--uuid:96ec3f73-ba4d-4d60-8c09-3645a3cb0b88+id=1
```

```
Content-ID: <http://tempuri.org/0>
Content-Transfer-Encoding: 8bit
Content-Type: application/xop+xml;charset=utf-8;type="application/soap+xml"

<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope" xmlns:u="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
 <s:Header>
  <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
   <u:Timestamp u:Id="_0">
    <u:Created>2012-09-26T19:38:38.193Z</u:Created>
    <u:Expires>2012-09-26T19:43:38.193Z</u:Expires>
   </u:Timestamp>
  </o:Security>
 </s:Header>
 <s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <COREEnvelopeBatchSubmissionResponse xmlns="http://www.caqh.org/SOAP/WSDL/CORERule2.2.0.xsd">
   <PayloadType xmlns="">X12_BatchReceiptConfirmation</PayloadType>
   <ProcessingMode xmlns="">Batch</ProcessingMode>
   <PayloadID xmlns="">3c7d1ea9-31d1-3bfe-b745-f7924c7940ac</PayloadID>
   <PayloadLength xmlns="">0</PayloadLength>
   <TimeStamp xmlns="">2012-09-26T19:38:37Z</TimeStamp>
   <SenderID xmlns="">77032</SenderID>
   <ReceiverID xmlns="">TP000001</ReceiverID>
   <CORERuleVersion xmlns="">2.2.0</CORERuleVersion>
   <ErrorCode xmlns="">Success</ErrorCode>
  </COREEnvelopeBatchSubmissionResponse>
 </s:Body>
</s:Envelope>
--uuid:96ec3f73-ba4d-4d60-8c09-3645a3cb0b88+id=1--
```

Figure 8: Sample MIME Multi-part form data BatchSubmissionResponseMessage

```
HTTP/1.1 202 Accepted
Content-Length: 2408
Content-Type: multipart/form-data; boundary=MISS_MIMEBOUNDRY
```

```
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="PayloadType"

X12_BatchReceiptConfirmation
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="ProcessingMode"

Batch
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="PayloadID"

f81d4fae-7dec-11d0-a765-00a0c91e6bf6
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="TimeStamp"

2022-08-30T10:20:34Z
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="SenderID"

77032
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="ReceiverID"

TP000001
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="CORERuleVersion"

2.2.0
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="ErrorCode"

Success
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="ErrorDescription"
```

```
--MISS_MIMEBOUNDRY--
```

3.7.5 BatchResultsRetrievalMessage

Download of batch result (i.e ASC X12, NCPD, MSCN, and REJ) files from the Gainwell Technologies batch repository requires a BatchResultsRetrievalMessage comprised of the HTTP/S envelope described in previous sections and the COREEnvelopeBatchResultsRetrieval envelope as shown in Table 21.

Batch results may be downloaded by specifying the appropriate payload type. If more than one batch result for the specified type exists on the server at the time of the request, then all the batch results for that specified type, up to the maximum download size, will be included in the response payload. Each file will be separated by an ASCII file separator character FS (0x01C).

Table 6: COREEnvelopeBatchResultsRetrieval description

Element	Description	Required
UserName	The submitter's Trading Partner ID. When using SOAP 1.2 messaging, the value is in the Username token of the security element within the SOAP header. Type: String Sample: TP000001	Y
Password	The submitter's password. When using SOAP 1.2 messaging, this value is in the Password token of the security element in the SOAP header. Password requirements can be found in the "Security Standards and Practices" section of this document. Type: String Sample: 1qaz@WSX	Y
PayloadType	The type of data being sent in the payload. A list of acceptable payload types can be found in Appedix B. Type: String Sample: X12_005010_Request_Batch_Results_271	Y
ProcessingMode	The mode in which the request should be processed. A list of acceptable processing modes can be found in Appedix B. Type: String Sample: Batch	Y
PayloadID	A unique identifier within the domain sending the request. A PayloadID may not be used more that once for a given ProcessingMode and SenderID combination. Type: UUID Sample: f81d4fae-7dec-11d0-a765-00a0c91e6bf6	Y

TimeStamp	The date and time, in UTC format, that the request was sent Type: DateTime Sample: 2022-08-30T10:20:34Z	Y
SenderID	The ID of the sender. For all requests, this must match the UserName. Type: String Sample: TP000001	Y
ReceiverID	The ID of receiver. For all requests, this value must be ‘77032’. Type: String Sample: 77032	Y
CORERulesVersion	The CORE envelope version being submitted. Mississippi Medicaid only supports CORERuleVersion 2.2.0. Type: String Sample: 2.2.0	Y

3.7.6 Sample XML and XML schema

Figure 9: Sample SOAP 1.2 MTOM BatchResultsRetrievalMessage

```
POST /Soap/COREService.svc/batch HTTP/1.1
Content-Type: multipart/related;type="application/xop+xml";start="<rootpart@soapui.org>";
        start-info="application/soap+xml; action=\"BatchResultsRetrievalTransaction\"";
        boundary="----=_Part_10_942815.1348749112477"
MIME-Version: 1.0
User-Agent: Jakarta Commons-HttpClient/3.1
Host: core.msxix.net
Content-Length: 1777



Content-Type: application/xop+xml;charset=UTF-8;
        type="application/soap+xml; action=\"BatchResultsRetrievalTransaction\""
Content-Transfer-Encoding: 8bit
Content-ID: <rootpart@soapui.org>

<soap:Envelope xmlns:cor=http://www.caqh.org/SOAP/WSDL/CORERule2.2.0.xsd”
    xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
```

```
 <soap:Header>
  <wsse:Security soap:mustUnderstand="true" xmlns:wsse="http://docs.oasis-
        open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
   <wsse:UsernameToken wsu:Id="UsernameToken-11" xmlns:wsu="http://docs.oasis-
         open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
    <wsse:Username>TP000001</wsse:Username>
    <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss
          -username-token-profile-1.0#PasswordText">1qaz@WSX</wsse:Password>
    <wsse:Nonce EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401
          -wss-soap-message-security-1.0#Base64Binary">
       /LYBa554cVZAjTAAZqyPEw==</wsse:Nonce>
    <wsu:Created>2012-09-27T12:31:52.461Z</wsu:Created>
   </wsse:UsernameToken>
  </wsse:Security>
 </soap:Header>
  <soap:Body>
   <cor:COREEnvelopeBatchResultsRetrievalRequest>
    <PayloadType>X12_005010_Request_Batch_Results_271</PayloadType>
    <ProcessingMode>Batch</ProcessingMode>
    <PayloadID>3c7d1ea9-41d1-3bfe-b746-f7924c7940ac</PayloadID>
    <PayloadLength>0</PayloadLength>
    <TimeStamp>2012-07-20T20:26:16.55Z</TimeStamp>
    <SenderID>TP000001</SenderID>
    <ReceiverID>77032</ReceiverID>
    <CORERuleVersion>2.2.0</CORERuleVersion>
    <CheckSum/>
    <Payload/>
   </cor:COREEnvelopeBatchResultsRetrievalRequest>
  </soap:Body>
</soap:Envelope>
------=_Part_10_942815.1348749112477--
```

Figure 10: Sample MIME Multi-part form data BatchResultsRetrievalRequest

```
POST /mime/COREservice.aspx HTTP/1.1
Host: core.msxix.net
```

```
Content-Length: 1385
Content-Type: multipart/form-data; boundary=MIME_BOUNDRY

--MIME_BOUNDRY
Content-Disposition: form-data; name="PayloadType"

X12_005010_Request_Batch_Results_271
--MIME_BOUNDRY
Content-Disposition: form-data; name="ProcessingMode"

Batch
--MIME_BOUNDRY
Content-Disposition: form-data; name="PayloadID"

f81d4fae-7dec-11d0-a765-00a0c91e6bf6
--MIME_BOUNDRY
Content-Disposition: form-data; name="TimeStamp"

2022-08-30T10:20:34Z
--MIME_BOUNDRY
Content-Disposition: form-data; name="UserName"

TP000001
--MIME_BOUNDRY
Content-Disposition: form-data; name="SenderID"

TP000001
--MIME_BOUNDRY
Content-Disposition: form-data; name="ReceiverID"

77032
--MIME_BOUNDRY
Content-Disposition: form-data; name="CORERuleVersion"

2.2.0
```

```
--MIME_BOUNDRY--
```

3.7.7 BatchResultsRetrievalResponseMessage

Table 7: COREEnvelopeBatchResultsRetrievalResponse description

Element	Description	Required
PayloadType	An appropriate response payload type. A list of possible response payload type can be found in Appedix B. Type: String Sample: X12_271_Response_005010X279A1	Y
ProcessingMode	The processing mode in which the request was submitted. Type: String Sample: Batch	Y
PayloadID	The PayloadID submitted in the request. Type: UUID Sample: f81d4fae-7dec-11d0-a765-00a0c91e6bf6	Y
PayloadLength	The length of data being sent in the Payload. Type: int Sample: 1024	PayloadLength
TimeStamp	The date and time, in UTC format, that the response was sent. Type: DateTime Sample: 2022-08-30T10:20:34Z	Y
SenderID	The ID of the sender. For all responses, this is ‘77032’. Type: String Sample: 77032	Y
ReceiverID	The ID of receiver. For all responses, this must match the Trading Partner ID that submitted the request. Type: String Sample: TP000001	Y
CORERulesVersion	The CORE envelope version being submitted. Mississippi Medicaid only supports CORERuleVersion 2.2.0. Type: String Sample: 2.2.0	Y

ErrorCode	The code of the error that occurred during message processing, not including ASC X12/NCPDP transactional errors. This field will have a value of 'Success' when a successful transmission occurs. A list of possible error codes may be found in "Error Handling" section of this document. Type: String Sample: Unauthorized	Y
ErrorMessage	A description of the error code. This field will be empty when a successful transmission occurs. Type: String Sample: Invalid username/password or not allowed to sign on from this location	Y
Checksum	A SHA-1 hash computed on the Payload contents. Type: String Sample: 6A3FE55946	Y
Payload	The transaction data being sent for processing. Type: String Sample: ISA*00* *00* *ZZ*TP000001…IEA*1*000000031~	Y

3.7.8 Sample XML and XML schema

Figure 11: Sample SOAP 1.2 MTOM BatchResultsRetrievalMessage

```
POST https://core.msxix.net/Soap/CoreService.svc/batch HTTP/1.1
Content-Type: multipart/related; type="application/xop+xml";start="<rootpart@soapui.org>";
        start-info="application/soap+xml; action=\"BatchResultsRetrievalTransaction\"";
        boundary="----=_Part_8_27294748.1348690941924"
MIME-Version: 1.0
User-Agent: Jakarta Commons-HttpClient/3.1
Host: localhost
Content-Length: 1778
Host: core.msxix.net


------=_Part_8_27294748.1348690941924
Content-Type: application/xop+xml; charset=UTF-8; type="application/soap+xml;
action=\"BatchResultsRetrievalTransaction\""
Content-Transfer-Encoding: 8bit
```

```
Content-ID: <rootpart@soapui.org>


<soap:Envelope xmlns:cor="http://www.caqh.org/SOAP/WSDL/CORERule2.2.0.xsd"
xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Header>
    <wsse:Security soap:mustUnderstand="true" xmlns:wsse="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
     <wsse:UsernameToken wsu:Id="UsernameToken-9" xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
      <wsse:Username>TP000001</wsse:Username>
      <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-
token-profile-1.0#PasswordText">1qaz@WSX</wsse:Password>
      <wsse:Nonce EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-
message-security-1.0#Base64Binary">ZMvBX+TrHjhHdaygt48Odg==</wsse:Nonce>
      <wsu:Created>2012-09-26T20:22:21.908Z</wsu:Created>
     </wsse:UsernameToken>
    </wsse:Security>
   </soap:Header>
   <soap:Body>
     <cor:COREEnvelopeBatchResultsRetrievalRequest>
       <PayloadType>X12_005010_Request_Batch_Results_271</PayloadType>
       <ProcessingMode>Batch</ProcessingMode>
       <PayloadID>3c7d1ea9-41d1-3bfe-b746-f7924c7940ac</PayloadID>
       <PayloadLength>0</PayloadLength>
       <TimeStamp>2012-07-20T20:26:16.55Z</TimeStamp>
       <SenderID>TP000001</SenderID>
       <ReceiverID>77032</ReceiverID>
       <CORERuleVersion>2.2.0</CORERuleVersion>
       <CheckSum/>
       <Payload/>
     </cor:COREEnvelopeBatchResultsRetrievalRequest>
   </soap:Body>
</soap:Envelope>
------=_Part_8_27294748.1348690941924—
```

Figure 12: Sample MIME Multi-part form data BatchResultsRetrievalMessage

```
HTTP/1.1 200 OK
Content-Length: 2408
Content-Type: multipart/form-data; boundary=XbCY

--XbCY
Content-Disposition: form-data; name="PayloadType"

X12_271_Response_005010X279A1
--XbCY
Content-Disposition: form-data; name="ProcessingMode"

Batch
--XbCY
Content-Disposition: form-data; name="PayloadID"

e51d4fae-7dec-11d0-a765-00a0c91e6da6
--XbCY
Content-Disposition: form-data; name="TimeStamp"

2022-08-30T10:20:34Z
--XbCY
Content-Disposition: form-data; name="SenderID"

77032
--XbCY
Content-Disposition: form-data; name="ReceiverID"

TP000001
--XbCY
Content-Disposition: form-data; name="CORERuleVersion"

2.2.0
--XbCY
Content-Disposition: form-data; name="ErrorCode"
```

```
Success
--XbCY
Content-Disposition: form-data; name="ErrorDescription"

--XbCY
Content-Disposition: form-data; name="Payload"

ISA*00*W2540582  *00*         *30*77032      *ZZ*TP000001… IEA*1*000000031~
--XbCY--
```

3.7.9 BatchSubmissionAckRetrievalMessage

The download of batch acknowledgement files from the batch repository requires a BatchSubmissionAckMessage comprised of the HTTP/S envelope described in previous sections and the COREEnvelopeBatchSubmissionAckRetrievalRequest envelope as shown in Table 23. Acknowledgements may be downloaded by specific type or all together using the appropriate payload type setting (e.g., X12_999_RetrievalRequest_005010X231A1 will only download 999s). If more than one acknowledgement exists on the server at the time of the request, all acknowledgements, up to the maximum download size, will be included in the payload. Each acknowledgment will be separated by an ASCII character (0x01C). If no acknowledgements exist at the time of the request, a payload type X12_005010_Response_NoBatchAckFile will be returned.

Table 8: COREEnvelopeBatchSubmissionAckRetrievalRequest description

Element	Description	Required
UserName	The submitter's Trading Partner ID. When using SOAP 1.2 messaging, this value is in the Username token of the security element within the SOAP header. Type: String Sample: TP000001	Y
Password	The submitter's password. When using SOAP 1.2 messaging, this value is in the Password token of the security element in the SOAP header. Password requirements can be found in the "Security Standards and Practices" section of this document. Type: String Sample: 1qaz@WSX	Y
PayloadType	The type of data being sent in the payload. A list of acceptable payload types can be found in Appedix B. Type: String Sample: X12_999_RetrievalRequest_005010X231A1	Y

ProcessingMode	The mode in which the request should be processed. A list of acceptable processing modes can be found in Appedix B. Type: String Sample: Batch	Y
PayloadID	A unique identifier within the domain sending the request. A PayloadID may not be used more that once for a given ProcessingMode and SenderID combination. Type: UUID Sample: f81d4fae-7dec-11d0-a765-00a0c91e6bf6	Y
TimeStamp	The date and time, in UTC format, that the request was sent. Type: DateTime Sample: 2022-08-30T10:20:34Z	Y
SenderID	The ID of the sender. For all requests, this must match the UserName. Type: String Sample: TP000001	Y
ReceiverID	The ID of receiver. For all requests, this value must be ‘77032’. Type: String Sample: 77032	Y
CORERulesVersion	The CORE envelope version being submitted. Mississippi Medicaid only supports CORERuleVersion 2.2.0. Type: String Sample: 2.2.0	Y

3.7.10 Sample XML and XML schema

Figure 13: Sample SOAP 1.2 MTOM BatchSubmissionAckRetrievalRequestMessage

```
POST https://core.msxix.net/Soap/CoreService.svc/batch HTTP/1.1
Content-Type: multipart/related; type="application/xop+xml";start="<rootpart@soapui.org>";
                start-info="application/soap+xml; action=\"BatchSubmitAckRetrievalTransaction\"";
                boundary="----=_Part_0_16643213.1348686053354"
MIME-Version: 1.0
Host: core.msxix.net
Content-Length: 1781

------=_Part_0_16643213.1348686053354
```

```
Content-Type: application/xop+xml; charset=UTF-8; type="application/soap+xml;
action=\"BatchSubmitAckRetrievalTransaction\""
Content-Transfer-Encoding: 8bit
Content-ID: <rootpart@soapui.org>

<soap:Envelope xmlns:cor="http://www.caqh.org/SOAP/WSDL/CORERule2.2.0.xsd"
          xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
  <soap:Header>
    <wsse:Security soap:mustUnderstand="true"
              xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-
1.0.xsd">
           <wsse:UsernameToken wsu:Id="UsernameToken-1"
                        xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd">
              <wsse:Username>TP000001</wsse:Username>
              <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
username-token-profile-
                        1.0#PasswordText">1qaz@WSX</wsse:Password>
              <wsse:Nonce EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
soap-message-security-1.0#Base64Binary">y0rowwkAzTndtPIfKzehRw==</wsse:Nonce>
              <wsu:Created>2012-09-26T19:00:53.337Z</wsu:Created>
           </wsse:UsernameToken>
         </wsse:Security>
  </soap:Header>
  <soap:Body>
<COREEnvelopeBatchResultsRetrievalResponse
xmlns="http://www.caqh.org/SOAP/WSDL/CORERule2.2.0.xsd">
      <PayloadType xmlns="">X12_271_Response_005010X279A1</PayloadType>
      <ProcessingMode xmlns="">Batch</ProcessingMode>
      <PayloadID xmlns="">3c7d1ea9-41d1-3bfe-b746-f7924c7940ac</PayloadID>
      <PayloadLength xmlns="">1000</PayloadLength>
      <TimeStamp xmlns="">2012-09-27T14:00:06Z</TimeStamp>
      <SenderID xmlns="">77032</SenderID>
      <ReceiverID xmlns="">TP000001</ReceiverID>
      <CORERuleVersion xmlns="">2.2.0</CORERuleVersion>
      <Payload xmlns="">
        <xop:Include href="cid:http%3A%2F%2Ftempuri.org%2F1%2F634843332091334609"
              xmlns:xop="http://www.w3.org/2004/08/xop/include"/>
```

```
      </Payload>
      <ErrorCode xmlns="">Success</ErrorCode>
    </COREEnvelopeBatchResultsRetrievalResponse>   </soap:Body>
</soap:Envelope>

--uuid:daa5d8d5-fa72-41f7-8183-00ac327dd579+id=2
Content-ID: <http://tempuri.org/1/634843332091334609>
Content-Transfer-Encoding: binary
Content-Type: application/octet-stream

[requested data here]
------=_Part_0_16643213.1348686053354—
```

Figure 14: Sample MIME Multi-part form data BatchSubmissionAckRetrievalRequestMessage

```
POST /mime/COREservice.aspx HTTP/1.1
Host: 77032:9443
Content-Length: 1385
Content-Type: multipart/form-data; boundary=MIME_BOUNDRY

--MIME_BOUNDRY
Content-Disposition: form-data; name="PayloadType"

X12_999_RetrievalRequest_005010X231A1
--MIME_BOUNDRY
Content-Disposition: form-data; name="ProcessingMode"

Batch
--MIME_BOUNDRY
Content-Disposition: form-data; name="PayloadID"

f81d4fae-7dec-11d0-a765-00a0c91e6bf6
--MIME_BOUNDRY
Content-Disposition: form-data; name="TimeStamp"

2022-08-30T10:20:34Z
```

```
--MIME_BOUNDRY
Content-Disposition: form-data; name="UserName"

TP000001
--MIME_BOUNDRY
Content-Disposition: form-data; name="SenderID"

TP000001
--MIME_BOUNDRY
Content-Disposition: form-data; name="ReceiverID"

77032
--MIME_BOUNDRY
Content-Disposition: form-data; name="CORERuleVersion"

2.2.0
--MIME_BOUNDRY--
```

3.7.11 BatchSubmissionAckRetrievalResponseMessage

Table 9: COREEnvelopeBatchSubmissionAckRetrievalResponse description

Element	Description	Required
PayloadType	An appropriate response payload type. A list of possible response payload type can be found in Appedix B. Type: String Sample: X12_005010_Response_Acks	Y
ProcessingMode	The processing mode submitted in the request. Type: String Sample: Batch	Y
PayloadID	The PayloadID submitted in the request. Type: UUID Sample: f81d4fae-7dec-11d0-a765-00a0c91e6bf6	Y
PayloadLength	The length of data being sent in the Payload. Type: int Sample: 1024	PayloadLength

TimeStamp	The date and time, in UTC format, that the response was sent. Type: DateTime Sample: 2022-08-30T10:20:34Z	Y
SenderID	The ID of the sender. For all responses, this must be '77032'. Type: String Sample: 77032	Y
ReceiverID	The ID of the receiver. For all responses, this must match the Trading Partner ID that submitted the request. Type: String Sample: TP000001	Y
CORERulesVersion	The CORE envelope version being submitted. Mississippi Medicaid only supports CORERuleVersion 2.2.0. Type: String Sample: 2.2.0	Y
ErrorCode	The code of the error that occurred during message processing, not including ASC X12/NCPDP transactional errors. This field will have a value of 'Success' when a successful transmission occurs. A list of possible error codes may be found in "Error Handling" section of this document. Type: String Sample: Unauthorized	Y
ErrorMessage	A description of the error code. This field will be empty when a successful transmission occurs. Type: String Sample: Invalid username/password or not allowed to sign on from this location	Y
Checksum	A SHA-1 hash computed on the Payload contents. Type: String Sample: 6A3FE55946	Y
Payload	The transaction data being sent for processing. When using SOAP 1.2 messaging payloads, data must be sent as a base64 encoded byte array in an MTOM attachement. If using MIME Multi-part form data payloads, data must be sent as plain text and in-line. Type: String Sample: ISA*00* *00* *ZZ*TP000001…IEA*1*000000031~	Y

3.7.12 Sample XML and XML schema

Figure 15: Sample SOAP 1.2 MTOM BatchSubmissionAckRetrievalResponseMessage

```
HTTP/1.1 200 OK
Content-Length: 2989
Content-Type: multipart/related; type="application/xop+xml";start="<http://tempuri.org/0>";
        boundary="uuid:a858b29e-bf60-433d-8aab-02646c19053c+id=4";start-
info="application/soap+xml"
Server: Microsoft-IIS/7.5
MIME-Version: 1.0
X-Powered-By: ASP.NET
Date: Wed, 26 Sep 2012 19:00:54 GMT



--uuid:a858b29e-bf60-433d-8aab-02646c19053c+id=4
Content-ID: <http://tempuri.org/0>
Content-Transfer-Encoding: 8bit
Content-Type: application/xop+xml;charset=utf-8;type="application/soap+xml"


<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope" xmlns:u="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
 <s:Header>
  <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
wss-wssecurity-secext-1.0.xsd">
    <u:Timestamp u:Id="_0">
     <u:Created>2012-09-26T19:00:54.112Z</u:Created>
     <u:Expires>2012-09-26T19:05:54.112Z</u:Expires>
    </u:Timestamp>
   </o:Security>
 </s:Header>
 <s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <COREEnvelopeBatchSubmissionAckRetrievalResponse
xmlns="http://www.caqh.org/SOAP/WSDL/CORERule2.2.0.xsd">
   <PayloadType xmlns="">X12_005010_Response_NoBatchAckFile</PayloadType>
   <ProcessingMode xmlns="">Batch</ProcessingMode>
   <PayloadID xmlns="">3c7d1ea9-31d1-3bfe-b745-f7924c7940ac</PayloadID>
   <PayloadLength xmlns="">1201</PayloadLength>
```

```
  <TimeStamp xmlns="">2012-09-26T19:00:53Z</TimeStamp>
  <SenderID xmlns="">77032</SenderID>
  <ReceiverID xmlns="">TP000001</ReceiverID>
  <CORERuleVersion xmlns="">2.2.0</CORERuleVersion>
  <Payload xmlns="">
   <xop:Include href="cid:http%3A%2F%2Ftempuri.org%2F1%2F634842648541127883"
xmlns:xop="http://www.w3.org/2004/08/xop/include"/>
  </Payload>
  <ErrorCode xmlns="">Success</ErrorCode>
 </COREEnvelopeBatchSubmissionAckRetrievalResponse>
</s:Body>
</s:Envelope>
--uuid:a858b29e-bf60-433d-8aab-02646c19053c+id=4
Content-ID: <http://tempuri.org/1/634842648541127883>
Content-Transfer-Encoding: binary
Content-Type: application/octet-stream

[requested data here]
--uuid:a858b29e-bf60-433d-8aab-02646c19053c+id=4—
```

NOTE: The actual payload is located in the MIME-attachment referenced by the include element.

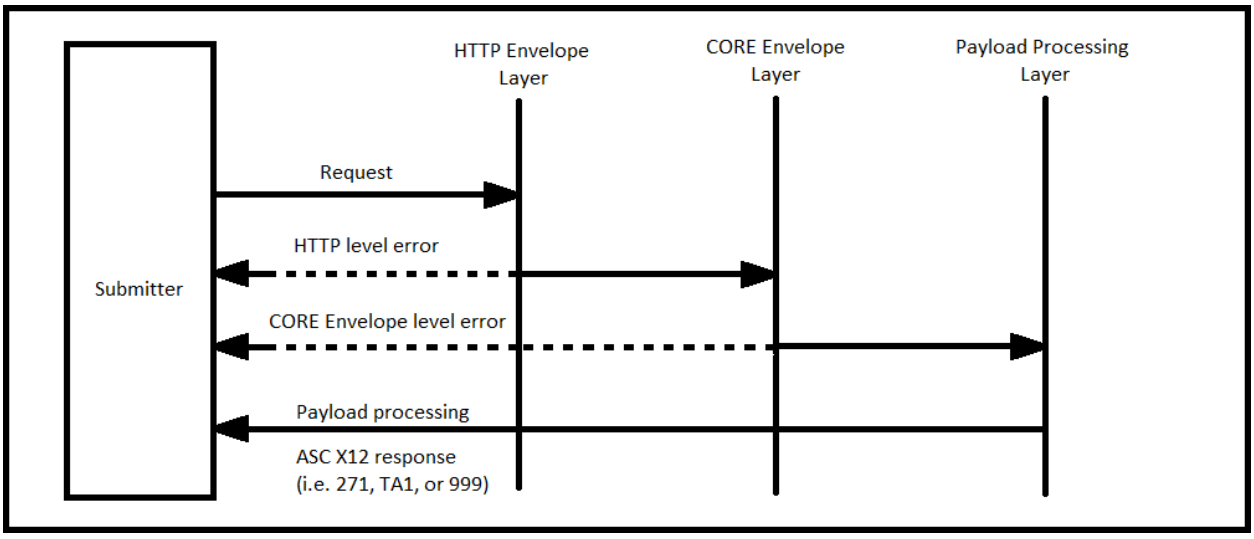
Figure 16: Sample MIME Multi-part form data RealTimeResponseMessage

```
HTTP/1.1 202 Accepted
Content-Length: 2408
Content-Type: multipart/form-data; boundary=MISS_MIMEBOUNDRY

--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="PayloadType"

X12_BatchReceiptConfirmation
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="ProcessingMode"

Batch
--MISS_MIMEBOUNDRY
```

```
Content-Disposition: form-data; name="PayloadID"

f81d4fae-7dec-11d0-a765-00a0c91e6bf6
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="TimeStamp"

2022-08-30T10:20:34Z
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="SenderID"

77032
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="ReceiverID"

TP000001
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="CORERuleVersion"

2.2.0
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="ErrorCode"

Success
--MISS_MIMEBOUNDRY
Content-Disposition: form-data; name="ErrorDescription"

--MISS_MIMEBOUNDRY--
```

4 Error Handling

The error handling layers described in this section are applicable to both HTTP/S envelope standards. Each request made to Mississippi Medicaid's CORE services goes through three (3) logical layers and if an error occurs at one of those logical layer (envelope only), then an error response will be returned. If an envelope passes validation and the payload makes it to the payload processor, then the appropriate response (ASC X12/NCPDP or Mississippi payload results) will indicate the outcome of processing.



NOTE: The dashed lines represent error messages returned if an error occurs at the corresponding processing layer. The solid lines represent request/response messages.

4.1 HTTP level status

The following HTTP status code situations apply to Batch uploads only:

- HTTP 202 Accepted
    - Indicates that the submitted request passed both HTTP envelope and CORE envelope validation. The request was accepted (uploaded) to the batch repository server for later processing. The response contains a CORE envelope error code of Success.

The following HTTP status codes situations apply to both Batch and RealTime:

- HTTP 200 OK
    - The submitted request passed HTTP envelope validation. However, CORE envelope validation may have failed. Check the CORE response envelope for possible errors. If no error is detected, the request was successful.

      NOTE: If requesting a batch upload, this error indicates that CORE envelope validation has failed. If batch upload is successful, a HTTP 202 will be returned.
- HTTP 400 Bad Request
    - HTTP envelope headers or MIME parts are malformed, check the response for likely causes.

- When using SOAP messaging and the response is empty. Most likely, the message could not be properly routed. Check the SOAP action headers.

- HTTP 403 Forbidden
  - The submitted request passed HTTP envelope validation but failed authentication of submitted credentials.
- HTTP 500 Internal Error
  - A message is received, but an internal process failed (e.g. batch or realtime services are down). Check the response for either: SOAP fault code “Receiver”, in the case of SOAP messaging, or a CORE envelope error code “Receiver”, in the case of MIME Multi-part messaging.
  - If no response message is contained in the HTTP response envelope, the message was most likely not received or was rejected at the server level. A likely cause of this is an invalid SOAP version (SOAP messaging only). Check the media type HTTP header for an “Unsupported media type” condition.
- HTTP 503 Service Unavailable
  - CORE web services are down.

4.1.1 CORE envelope status/error codes and descriptions

Element	Description
Success	The message was processed successfully.
FilesPending	set on transaction or ack download responses when more file are available for download for the type specified in the request.
<FieldName>_Illegal	The value provided for <FieldName> is valid.
<FieldName>_Required	The field <FieldName> is required but was not provided.
<FieldName>_NotUnderstood	The field <FieldName> is not understood at the receiver. In the case of SOAP, this error is returned as a “NotUnderstood SOAP” fault.
VersionMismatch	The version on CORE envelope is not acceptable.
UnAuthorized	The username/password supplied is not authorized
ChecksumMismatched	The checksum computed on the payload by the receiver did not match the checksum sent by the client.
Sender	The CORE envelope sent by the sender contained data that was legal but could not be processed for some reason (e.g., duplicate PayloadID, password strength when changing password, use of unsupported CORE web methods).
Receiver	The message could not be processed for some reason attributable to the receiver (e.g., upstream process not available). In the case of SOAP, this error will be returned in a SOAP fault with fault code “Receiver”.

5 SFTP (FTP over SSH) Specification

This section covers the most frequently used SFTP commands. An SFTP connection can be established using “fts.msxix.net” port 22.

NOTE: Users are required to change passwords on first login; you must log in to https://fts.msxix.net/ to change your password.

5.1.1 Uploading files

To upload files via SFTP, connect to “fts.msxix.net” using an Trading Partner ID, password and SSH Key. To upload a file, navigate to /home/Production/edi/<Trading Partner ID>/Inbox and issue the “Put” command and the name of the file to be uploaded.

Example: put foo.txt

5.1.2 Directory listing

To locate available response files via SFTP, connect to “fts.msxix.net” using an Trading Partner ID, password and SSH Key. Then, navigate to /home/Production/edi/<Trading Partner ID>/Outbox and execute a “dir” command.

Example: dir

5.1.3 Downloading files

To download files via SFTP, connect to “fts.msxix.net” using an Trading Partner ID, password and SSH Key. Then, navigate to “cd” to /home/Production/edi/<Trading Partner ID>/Outbox and retrieve the file using the “get” command and the FileName or Field.

Example: get foo.txt

6 Security Standards and Practices

In order for Gainwell Technologies to guarantee a safe and stable working environment, the following security standards and practices have been established. These measures protect users and Gainwell Technologies from potential threats.

6.1 Password creation requirements

Passwords must contain:

- A minimum of 8 alpha-numeric-special characters
- At least 1 upper and 1 lower case alpha character
- At least 1 number
- At least 1 special character

Passwords **cannot**:

- Be similar to username (Trading Partner ID)
- Use any common 'dictionary' words
- Use previous six (6) passwords

6.2 New File Transfer System client passwords

Any new client to the File Transfer System will be given a temporary password that meets the password requirement criteria. This password will be a one-time-use only password that must be changed upon first login. The new password should be of the client's choosing and must adhere to the required password criteria listed above.

6.3 Password aging

- Passwords expire after 60 days.
- Each method of interaction with File Transfer System, excluding SSH protocol, has its own procedures for changing passwords as deemed necessary by the system. Each of these procedures can be reviewed in the following subsections.

6.3.1 Website password change procedures

The File Transfer System website has a "warning period" for notifying users prior to password expiration. When a user logs in during this period, they will be prompted to change their password. If the user fails to change their password during the "warning period" and allows it to exceed its 60-day life span, the account will be locked, and the user will need to contact the EDI Help Desk for further assistance.

NOTE: Passwords may be changed at any time by logging into the Mississippi Medicaid secure website at https://fts.msxix.net/.

6.3.2 Web service password change procedures

File Transfer System clients using the web service interface will not be given a warning period prior to their password expiration. They will only be notified that their password has aged on the 60th day. At

this time, the user will not be allowed to perform any upload, download, directory, or delete transactions until the password has been changed. If the user fails to change their password and the account password is allowed to exceed a 60 day life span, the account will be locked out, and the user will need to contact the EDI Help Desk for further assistance.

Table 10: File Transfer System supports the following FTP commands for the purposes of integrity checking:

Command	Description
INTEGRITY type	Enables integrity checking on STOR and RETR. For type L, use “lump mode” data stream which compresses data on the fly and include a SHA1 hash of the file for integrity verification. For type H, “hash mode”, the data stream is standard, and the client is responsible for checking the SHA1 hash for integrity verification. Type N turns integrity checking off.
XSHA1 filename	Returns the SHA1 hash of the file named, usually the most recently transferred file.
HASH OK/BAD	Notifies File Transfer System FTP, after a STOR or RETR, that the client has verified the file SHA1 hash and that it has matched (or not matched) the one passed in the data stream in lump mode, or returned by the XSHA1 command.

Table 11: File Transfer System supports the following FTP command for the purpose of changing a password:

Command	Description
CHGPW oldpassword newpassword newpassword	Sends a password change request to File Transfer System for the logged-on user. The old password is rechecked and the new passwords must match and meet strength requirements for the site.

File Transfer System **unsupported sftp commands:** ACCT, SMNT, REIN, STOU, ALLO, ABOR, SITE

7 Appendix A: CORE Service Types

7.1 Processing Modes

- Batch
- RealTime

7.2 RealTime Payload Types

Request PayloadTypes

- X12_270_Request_005010X279A1
- X12_276_Request_005010X212
- X12_278_Request_005010X217
- NCPDP_B1_Request_008001D012
    - Payload Encrypted Base 64
- NCPDP_B2_Request_008001D012
    - Payload Encrypted Base 64
- NCPDP_B3_Request_008001D012
    - Payload Encrypted Base 64

Response PayloadTypes

- X12_271_Response_005010X279A1
- X12_277_Response_005010X212
- X12_278_Response_005010X217
- NCPDP_B1_Response_008001D012
    - Payload Encrypted Base 64
- NCPDP_B2_Response_008001D012
    - Payload Encrypted Base 64
- NCPDP_B3_Response_008001D012
    - Payload Encrypted Base 64

7.3 Batch Payload Types

Batch Upload Request PayloadTypes

- X12_270_Request_005010X279A1
- X12_276_Request_005010X212
- X12_278_Request_005010X217
- X12_837_Request_005010X222A1
- X12_837_Request_005010X223A2
- X12_837_Request_005010X224A2

- NCPDP_B1_Request_008001D012
    - Payload Encrypted Base 64
- NCPDP_B2_Request_008001D012
    - Payload Encrypted Base 64
- NCPDP_B3_Request_008001D012
    - Payload Encrypted Base 64
- X12_999_SubmissionRequest_005010X231A1
    - This payload type only logs that the submitter is acknowledging receipt of a requested download, if 999 data is sent in the payload is will not be uploaded to Mississippi Medicaids' batch repository server or processed.

Batch Upload ResponseType

- X12_BatchReceiptConfirmation
    - For the submission of all batch upload payload types except X12_999_SubmissionRequest_005010X231A1
- X12_Response_ConfirmReceiptReceived
    - For the submission of an X12_999_SubmissionRequest_005010X231A1

Batch Download Request PayloadTypes

- X12_005010_Request_Batch_Results_271
- X12_005010_Request_Batch_Results_277
- X12_278_Request_Batch_Results_005010X217
- X12_835_Request_005010X221A1
- X12_999_RetrievalRequest_005010X231A1
- X12_TA1_RetrievalRequest_00501X231A1
- X12_005010_Request_Acks
    - Downloads all acknowledgment types 999 and TA1

Batch Download Response PayloadTypes

- X12_271_Response_005010X279A1
- X12_277_Response_005010X212
- X12_278_Response_005010X217
- X12_835_Response_005010X221A1
- X12_999_Response_005010X231A1
- X12_TA1_Response_00501X231A1
- X12_005010_Response_Acks
    - Mixed acknowledgements
- X12_005010_Response_NoBatchAckFile

    - No acknowledgements available
- X12_005010_Response_NoBatchResultsFile
    - No batch files for requested type available for download

8 Appendix B: CORE sample programs

Submitting a batch 270 (this could have been any acceptable Mississippi Medicaid batch transaction)

NOTE: All sample programs illustrated are done in vs2010 C#; using service reference generation tool pointed at the following address https://core.msxix.net/soap/coreservice.svc?wsdl . These samples are purely for illustration purposes and not meant to be production quality.

```
class Program
  {
    static void Main( string[] args )
    {
      String ruleVersion = "2.2.0";
      String senderId = "TP000001";
      String receiverId = "77032";
      String batchProcMode = "Batch";

      var client = new BatchClient(); // used for batch transactions

      client.ClientCredentials.UserName.UserName = senderId;
      client.ClientCredentials.UserName.Password = "Mypassword";

      var submitTrans = new COREEnvelopeBatchSubmission()
        {
          PayloadType = "X12_270_Request_005010X279A1",
          ProcessingMode = batchProcMode,
          PayloadID = Guid.NewGuid().ToString(),
          CORERuleVersion = ruleVersion,
          SenderID = senderId,
          ReceiverID = receiverId,
          TimeStamp = DateTime.UtcNow.ToString( "yyyy-MM-ddThh:mm:22Z" ),
          // complete transaction omitted for brevity
          Payload = System.Text.Encoding.ASCII.GetBytes( "ISA....IEA...~" ),
          PayloadLength = "ISA....IEA...~".Length
        };

      var submitResp = client.BatchSubmitTransaction( submitTrans );
```

```
if ( submitResp.ErrorCode.Equals( "Success" ) )
{
   // successful status code returned
   if ( submitResp.PayloadType.Equals( "X12_BatchReceiptConfirmation" )
   {
      // expected batch payload type returned
            // process the response
   }
}

// wait for at least 15 minutes (maybe longer) to retrieve batch acknowledgements
// MS Medicaid only returns negative acknowledgements
var acksTrans = new COREEnvelopeBatchSubmissionAckRetrievalRequest()
{
   PayloadType = "X12_005010_Request_Acks",  // retrieve available 999 and TA1
   ProcessingMode = batchProcMode,
   PayloadID = Guid.NewGuid().ToString(),
   CORERuleVersion = ruleVersion,
   SenderID = senderId,
   ReceiverID = receiverId,
   TimeStamp = DateTime.UtcNow.ToString( "yyyy-MM-ddThh:mm:22Z" )
};

var acksResp = client.BatchSubmitAckRetrievalTransaction( acksTrans );

if ( acksResp.ErrorCode.Equals( "Success" ) )
{
   // successful status code returned

   if ( submitResp.PayloadType.Equals( "X12_005010_Response_Acks" ) )
   {
      // do something with acks

   }
   else if ( submitResp.PayloadType.Equals( "X12_005010_Response_NoBatchAckFile" ) )
```

```
    {
        // no acks available at this time
        // try again later.
    }
}


// wait a minimum of 1 hr retrieve batch results that did not generate a negative
// acknowledgement

var resultsTrans = new COREEnvelopeBatchResultsRetrievalRequest()
{
    PayloadType = "X12_005010_Request_Batch_Results_271", // retrieve batch 271's
    ProcessingMode = batchProcMode,
    PayloadID = Guid.NewGuid().ToString(),
    CORERuleVersion = ruleVersion,
    SenderID = senderId,
    ReceiverID = receiverId,
    TimeStamp = DateTime.UtcNow.ToString( "yyyy-MM-ddThh:mm:22Z" )
};

var resultsResp = client.BatchResultsRetrievalTransaction( resultsTrans );

if ( resultsResp.ErrorCode.Equals( "success", StringComparison.CurrentCultureIgnoreCase ) )
{
    // successful status code returned

    if ( resultsResp.PayloadType.Equals( "X12_271_Response_005010X279A1" ) )
    {
        if ( resultsResp.Payload != null && resultsResp.Payload.Length > 0 )
        {
            // payload available
            // split into individual files; each file separated by 0x01C
            var payload = Encoding.ASCII.GetString( resultsResp.Payload ).Split( new char[] {
                                (Char)0x01C }, StringSplitOptions.RemoveEmptyEntries );
```

```
                // check if more files still waiting download
                if ( resultsResp.ErrorCode.Equals( "FilesPending" ) )
                {
                    // more files waiting download get them, re-using previous transaction except
                    // payload id and timestamp
                    resultsTrans.PayloadID = Guid.NewGuid().ToString();
                    resultsTrans.TimeStamp = DateTime.UtcNow.ToString( "yyyy-MM-ddThh:mm:22Z" );
                    resultsResp = client.BatchResultsRetrievalTransaction( resultsTrans );

                    // process these files
                }

            }
        }
    }
}
```

Submitting a real-time 270 (this could have been any acceptable Mississippi Medicaid realtime transaction).

NOTE: All sample programs illustrated are done in vs2010 C#; using service reference generation tool pointed at the following address https://core.msxix.net/soap/coreservice.svc?wsdl . These samples are purely for illustration purposes and not meant to be production quality.

```
class Program
 {
    static void Main( string[] args )
    {
        String ruleVersion = "2.2.0";
        String senderId = "TP000001";
        String receiverId = "77032";
        String procMode = "RealTime";

        // used for realtime or admin transactions
        var client = new RealtimeClient();
```

```
        client.ClientCredentials.UserName.UserName = senderId;
        client.ClientCredentials.UserName.Password = "Mypasswor";


    var realTimeTrans = new COREEnvelopeRealTimeRequest()
        {
            PayloadType = "X12_270_Request_005010X279A1",
            ProcessingMode = procMode,
            PayloadID = Guid.NewGuid().ToString(),
            CORERuleVersion = ruleVersion,
            SenderID = senderId,
            ReceiverID = receiverId,
            TimeStamp = DateTime.UtcNow.ToString( "yyyy-MM-ddThh:mm:22Z" ),
            Payload = "ISA....IEA...~", // complete transaction omitted for brevity
        };

    var realTimeResp = client.RealTimeTransaction( realTimeTrans );

    if( realTimeResp.ErrorCode.Equals("Success") )
    {
         if( realTimeResp.PayloadType.Equals( "X12_271_Response_005010X279A1" ) )
          {
              // correct payload type
              if( realTimeResp.Payload.Length > 0 )
              {
                  // process response payload
              }
          }
     }
   }
}
```