File Transfer Service User Guide

A Guide to Submitting batches through eMedNY FTS

STATE OF NEW YORK
DEPARTMENT OF HEALTH

March 30, 2020

Table of Contents

1 Introduction

Simple Object Access Protocol (SOAP) is an XML based protocol which enables applications to exchange information over Hyper Text Transfer Protocol (HTTP) and other protocols. Primarily, SOAP provides a way to communicate between applications running on different operating systems, with different technologies and programming languages. Simply stated, SOAP is a protocol for accessing a Web Service. There are two web services currently supported by eMedNY

- File Transfer System (for batch file submissions)
- eMedNY Meds History Service (for Real Time Eligibility Verification and Meds History Transactions)

This document concerns the File Transfer Service (FTS,) or Batch File Transaction service, of New York State Medicaid. The File Transfer Service SOAP web service allows users to submit a file consisting of a batch of transactions of a certain type to eMedNY for processing in batch mode. A list of supported file formats for inbound and outbound transactions is in section 4. The file is a XML-formatted file containing the transaction in their native formats. The input file is stored in the user's eXchange inbox along with the output files that result from the delayed batch processing of the input file. Using the web service, the user can also view a list of files in his mailbox and retrieve any file on the list representing the input file or a processed response file. Each of the afore-mentioned functions is invoked by a separate web service operation (see Web Service Operations in part 3.)

Trading Partners may use SOAP and its associated Service Oriented Architecture (SOA) for exchange of batch files with eMedNY. This process is initiated by an ePACES System Administrator, who can request, receive, and revoke eMedNY issued User Certificates. The Key and Certificate Management User Guide and eMedNY X509 Certificate Request User Certificate Guide gives instructions for establishing an eMedNY SOAP Administrator and to create an eMedNY SOAP User Certificate.

SOAP inbound data-streams will be authenticated by verifying the User Authentication Certificate. Each ePACEs Administrator is allowed only one User Certificate. The User Certificate will have a **six month life span**.

Important Note: It is solely the responsibility of the submitter or user, who wishes to utilize the eMedNY SOAP submission architecture, to develop or create their own SOAP compliant application. eMedNY will in no way support the end user SOAP Application, therefore it is strongly recommended that the Trading partner take appropriate action to have available technical support.

2 Requirements

The following requirements need to be met before the web service can be used.

2.1 Exchange mailboxes

The user has to have a pre-established exchange mailbox in eMedNY. If an ePACES account was used to request the original certificate as outlined in the eMedNY X509 Certificate Request User Guide , then see the following link for information on eMedNY eXchange.

If an FTP account was used to request the original Certificate then an eXchange inbox was created with the same userID and password in that process.

2.2 Security

The information exchange between user and eMedNY using any of the web service operations must be protected by encryption and the user's identity must also be authenticated by eMedNY.

2.2.1 Two-way SSL

The encryption and authentication is achieved using two-way SSL.

This requires that the user apply for an eMedNY-signed Client Certificate via a process outlined in the document eMedNY X509 Certificate Request User Guide. This certificate will be installed on the user's system.

The user can also be given the eMedNY server certificate which the user can add to the client system's certificate store as well. Please email emednyproviderservices@gdit.com for the server certificate.

These two certificates are used for the purpose of encrypting the information exchanged between the client and eMedNY, and also to allow mutual authentication between eMedNY and the user.

2.2.2 Username and Password

Multiple users can share the same user certificate for the File Transfer Service if the users fall under the same security administration of the organization requesting the eMedNY-signed User Certificate. In order to protect the confidentiality of the information in each user's mailbox, a second authentication factor is implemented for the File Transfer Service, wherein every invocation of a web service operation must include the user's mailbox username and password as well. In this way only the knowledge of a user's mailbox credentials (username and password) will allow access to that user's mailbox. Without this credential, the web service operation will fail.

3 Web Service Operations

The eMedNY system is available 24 X 7 for submissions. Note that the eMedNY FTS submissions will have a 28 day file retention period, which is consistent with other batch access methods available to NYS Medicaid.

The following operations are provided by the File Transfer Service. Any operation that is invoked unsuccessfully will result in a SOAP fault being returned. In Section 3.4 there is a list of possible SOAP faults.

3.1.1 sendData

This operation sends a file to eMedNY. In addition to the Username and Password for the user's mailbox, the operation requires the file type to be selected from one of these file transaction types:

ISA – for all x12 transactions
NCP – for NCPDP transactions

The payload for the sendData request is the transactions sent in native format (i.e. 270, 278 etc,) encoded in base-64 binary for SOAP transmission. The allowed inbound transactions are listed in Section 4.1 Inbound Transactions.

A sample SOAP XML message for the sendData operation is given below:

3.1.1.1 Sample sendData Request & Response

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:fts="http://org/emedny/fts/">
  <soapenv:Header/>
  <soapenv:Body>
    <fts:sendData>
      <fts:input>
<fts:transData>SVNBKj……………etc……..yflNWMiowNDMwKkhDfDk3NTM3KjkzLjEqVU4qMX5EVFAqNDcyKkQ4KjIwMDcw
OTA0flNFKjM0NDgyKjM2MDgxfkdFKjEqMzYwODB+SUVBKjEqMDAwMDM2MDc4fg==</fts:transData>
        <fts:userName>YOUR_UID</fts:userName>
        <fts:passWord>YOUR_PASSWORD</fts:passWord>
        <fts:transType>ISA</fts:transType>
      </fts:input>
    </fts:sendData>
  </soapenv:Body>
</soapenv:Envelope>
```

The information payload between <fts:transData> and </fts:transData> is the base64 encoded form of an X12 270 transaction (See References for links to Base 64 Encoding.) As the payload can be huge, SOAP allows a more efficient form of physical transmission known as MTOM (See References for links to MTOM), wherein the payload between

<fts:transData> and </fts:transData> is actually sent outside the <soap:Envelope> as an attachment. The mode of transmission (inline or MTOM) is set by the client. The server is configured to handle either.

A successful response to this would look like the following:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <sendDataResponse xmlns:xmime=http://www.w3.org/2005/05/xmlmime  xmlns="http://org/emedny/fts/">
      <output>
        <fileName>YOUR_UID__-111216161620-00-ISA-00-.x12</fileName>
      </output>
    </sendDataResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Note that the fileName returned follows the convention:

USERNAME-TIMEDATE-SEQ-TYPE-INDICATOR.soa

While an error would be encapsulated in a SOAP Fault

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Body>
    <env:Fault>
      <faultcode>env:Server</faultcode>
      <faultstring>Internal Error (from server)</faultstring>
    </env:Fault>
  </env:Body>
</env:Envelope>
```

The faultcode and faultstring are defined in Section 3.4. The faultcode is primarily of use in machine to machine processing whereas the fault string is intended for human users. Automated corrective actions can be made based on analysis of the fault code in some cases (e.g. an automatic resend if there is a connection failure) but in other cases, human intervention is required to determine further course of action.

3.2 getFileNames

This operation allows the user to retrieve a list of filenames in the user's mailbox.

3.2.1 Sample getFileNames Request & Response

A sample getFileNames request follows:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:fts="http://org/emedny/fts/">
  <soapenv:Header/>
  <soapenv:Body>
    <fts:getFileNames>
      <fts:input>
             <fts:userName>YOUR_UID</fts:userName>
            <fts:passWord>YOUR_PASSWORD</fts:passWord>
      </fts:input>
    </fts:getFileNames>
  </soapenv:Body>
</soapenv:Envelope>
```

The response returns the 500 most recent filenames available for download in FTS inbox for that UserID. Responses become unavailable for download after four weeks. A sample getFileNames response follows:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <getNamesResponse xmlns:xmime="http://www.w3.org/2005/05/xmlmime" xmlns="http://org/emedny/fts/">
      <output>
        <fileNames>YOUR_UID__-111216161620-00-ISA-00-.x12</fileNames>
        <fileNames>YOUR_UID__-111114142242-00-ISA-00-.x12</fileNames>                    ......etc.......
      </output>
    </getNamesResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

3.3 getFile

This operation allows the user to retrieve the file contents of one of the files showing in the previous getNames Response list of filenames. Here we retrieve the file that was just sent.

3.3.1 Sample getFile Request & Response

A sample getFile request follows:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:fts="http://org/emedny/fts/">
  <soapenv:Header/>
  <soapenv:Body>
    <fts:getFile>
```

```
      <fts:input>
         <fts:userName>YOUR_UID</fts:userName>
        <fts:passWord>YOUR_PASSWORD</fts:passWord>
        <fts:fileName> YOUR_UID__-111216161620-00-ISA-00-.x12</fts:fileName>
      </fts:input>
    </fts:getFile>
  </soapenv:Body>
</soapenv:Envelope>
```

A sample getFileNames response follows:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <getResponse xmlns:xmime="http://www.w3.org/2005/05/xmlmime" xmlns="http://org/emedny/fts/">
      <output>
        <fileData>
          <xop:Include href="cid:urn:uuid:4FE35C12AC4A5630DC1324071297656@apache.org"
xmlns:xop="http://www.w3.org/2004/08/xop/include"/>
        </fileData>
      </output>
    </getResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Notice that the fileData has an <xop:Include> the contents of which has a href value (a CID, or content ID, which points to the attachment which has the data.) This "<xop:Include>" tag indicates that the content has been sent back using MTOM (because there is a lot of content and the server has determined that it is more efficient to send it back using MTOM.) Your client software for handling SOAP web services should have the necessary ability to retrieve the referenced attachment in the message that is sent by MTOM.

The raw HTTP message your client SOAP software sees after decryption is illustrated here:

```
HTTP/1.1 200 OK
X-Backside-Transport: OK OK
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: multipart/related; boundary="MIMEBoundaryurn_uuid_4FE35C12AC4A5630DC1324071297643"; start-info="text/xml"; type="text/xml"; start="<0.urn:uuid:4FE35C12AC4A5630DC1324071297644@apache.org>"
Content-Language: en-US
Date: Fri, 16 Dec 2011 21:34:56 GMT
Server: WebSphere Application Server/7.0
X-Client-IP: XX.XX.XXX.XX
```

```
--MIMEBoundaryurn_uuid_4FE35C12AC4A5630DC1324071297643
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: binary
Content-ID: <0.urn:uuid:4FE35C12AC4A5630DC1324071297644@apache.org>

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"><soapenv:Body><getResponse
xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
xmlns="http://org/emedny/fts/"><output><fileData><xop:Include
href="cid:urn:uuid:4FE35C12AC4A5630DC1324071297656@apache.org"
xmlns:xop="http://www.w3.org/2004/08/xop/include"/></fileData></output></getResponse></soapenv:Body></soape
nv:Envelope>
--MIMEBoundaryurn_uuid_4FE35C12AC4A5630DC1324071297643
Content-Type: application/octet-stream
Content-Transfer-Encoding: binary
Content-ID: <urn:uuid:4FE35C12AC4A5630DC1324071297656@apache.org>
ISA*00*          *00*          *ZZ*JX6            *ZZ*EMEDNYBAT      *080811*0915*^*00501 ......etc...
```

The last line giving the X12 content of the file you retrieved

3.4 SOAP Faults

Errors encountered in any of the above SOAP operations will be returned as SOAP faults

The fault consists of a numeric error code and a descriptive error message. The former can be used to enable automated processing of errors based on selected error codes while the latter are for those errors that need human intervention.

The following is a list of errors returned by the web service:

Error Code	Error Message
1000	Missing User Name
1001	Invalid User Name
1002	Missing Transaction Type
1003	Invalid Credentials
2000	Missing File Name

2001	Error Writing File
2003	File Does Not Exist
3000	Internal IO Error
3001	No Mail Box For User
5000	Internal Server Error
5001	Illegal Operation
5002	Database Error
5003	Authentication System Error

4 Supported File Formats for Batch File Transactions

The x12 coding of batch files sent through the File Transfer Service are exactly the same as for other transmission methods. The X12 TR3's (Implementation Guides) that detail the full requirements for transactions sent and received by eMedNY can be licensed at http://store.x12.org/store/. Please refer to the eMedNY Transaction Information Standard Companion Guide for additional information that is specific to New York State Medicaid.

4.1 INBOUND Transactions:

- X12N/005010X279 Health Care Eligibility Benefit Inquiry (270)
- X12N/005010X212 Health Care Claim Status Request (276)
- X12N/005010X217 Health Care Services Review Request for Review (278)
- X12N/005010X220 Benefit Enrollment and Maintenance (834)
- X12N/005010X231A1 999 Submission Request (999 Response to an outbound 834)
- X12N/005010X223A2 Health Care Claim Institutional (837)
- X12N/005010X222A1 Health Care Claim Professional (837)
- X12N/005010X224A2 Health Care Claim Dental (837)
- NCPDP Version 5.1 - National Council for Prescription Drug Programs (REQUEST)

4.2 OUTBOUND Transactions:

- X12C/005010X231 Implementation Acknowledgment For Health Care Insurance (999)
- X12N/005010X279 Health Care Eligibility Benefit Response (271)
- X12N/005010X212 Health Care Claim Status Response (277)
- X12N/005010X214 Health Care Claim Acknowledgment (277)
- X12N/005010X217 Health Care Services Review Response (278)
- X12N/005010X220 Benefit Enrollment and Maintenance (834)
- X12N/005010X221A1 Health Care Claim Payment/Advice (835)
- X12N/005010X218 Payroll Deducted and Other Group Premium Payment for Insurance Products (820)
- NCPDP Version 5.1 - National Council for Prescription Drug Programs (RESPONSE)

5 URL For Batch File Transmissions

The URL and port to transmit batch files through eMedNY SOAP File Transfer Services is:

https://fts.emedny.org:8443/eMedNYServices/FTService/FTService

This link is only accessible if you have an eMedNY user certificate.

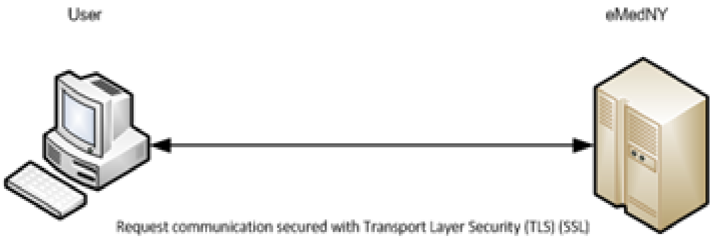
6 Graphical Depiction of the Process flow

6.1 eMedNY Certificate Request Process

1. User creates Keystore – using Keytool or other mechanism.

2. User Creates Certificate Signing Request (CSR) for eMedNY Certificate Authority (CA) – Using Keytool or other mechanism.

3. User accesses eMednyCA application for certificate generation, using CSR.


User
eMedNY
Request communication secured with Transport Layer Security (TLS) (SSL)

4. eMedNY accepts CSR from "qualified" requesting user.

5. eMedNY staff review and approve/disprove certificate request.

6. eMedNY Integrated Cryptographic Services Facility (ICSF) generates eMedNY certificate for use with eMedNY web services.

7. eMedNY ICSF posts eMedNY certificate, user ID, and password for requesting user to retrieve.

8. User retrieves eMedNY certificate, UserID, and Password.

9. User imports certificate into Keystore, and records User ID and password information associated with eMedNY web services.

10. User now has eMedNY certificate, User ID, and password available for eMedNY web service access.

Please Note: The entity wishing to access eMedNY using the certificate is responsible for purchasing or developing all necessary applications required to leverage the eMedNY certificate, User ID, and password information.

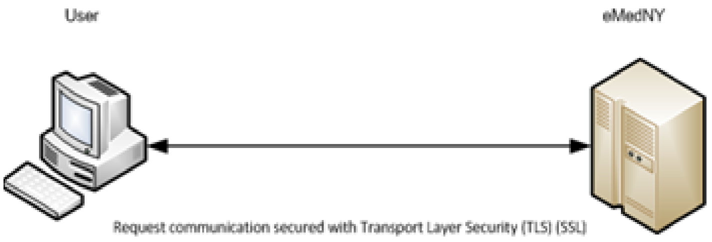
6.2 eMedNY Web Service Access

11. User imports eMedNY Web Service Definition Language (WSDL) and XML schemas into their application.
    - File Transfer Service (FTS)

12. User makes certificate Keystore (eMedNY Certificate) and eMedNY user credentials (ID and password) available to their application.

13. User application connects to eMedNY Enterprise Service Bus (ESB)
    - Tunnel encryption (TLS/SSL) negotiated between user and eMedNY ESB

14. User application formats the XML request message required by FTS web service according to its wsdl, (which defines a username and password to be part of the XML message) and places it in the body of the SOAP request


User
eMedNY
Request communication secured with Transport Layer Security (TLS) (SSL)

16. eMedNY processes the SOAP request, authenticates the contained credentials, and sends a SOAP response.

15. User application communicates with eMedNY web server using 2-wayTransport Layer Security.

17. User receives response and applies MLS to decrypt SOAP body and verify Signature of response from eMedNY

7 SOAP and the eMedNY Dashboard

Files sent through the File Transfer Service can be viewed in the eMedNY Dashboard. The Dashboard is a tool for viewing files sent to eMedNY, their status, and any errors they may have encountered. A detailed guide to the Dashboard can be found here.

Users looking for information on their files sent via FTS should input their Username, password, and select "FTS" as the submission source.

8 Testing

At this time, the File Transfer Service cannot access the Provider Test Environment (PTE). Any files sent with a “T” indicator in the ISA15 will result in a rejection sent via an eMedNY Front-End Rejection “F” file.

Receipt of this “F” file rejection can be considered a successful test of connectivity through SOAP FTS. If providers wish to test ISA formatting of their files, they will need to use eXchange or FTP to send these files.

9 Additional Tools & Information

9.1 Contact Information

If you experience issues with submission, you may contact the eMedNY Call Center at (800) 343-9000 or email emednyproviderservices@gdit.com

9.2 SOAP Web Links

http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

http://www.w3.org/TR/soap12-part1/#intro

http://www.w3.org/TR/soap12-mtom/

9.3 Third Party SOAP Clients

9.3.1 SOAPSonar

http://www.crosschecknet.com/products/soapsonar.php

http://www.crosschecknet.com/doc/releasenotes.htm

9.3.2 SOAPUI

http://www.soapui.org/

9.4 Requirements for CORE Compliance

http://www.caqh.org/benefits.php

9.5 FTP and ePACES Access Information & FAQ

Note: Trading Partners who are currently not enrolled in ePACES, must successfully enroll in ePACES in order to obtain a SOAP Administrator User ID & Password.

http://www.emedny.org/selfhelp/index.html

http://www.emedny.org/selfhelp/epaces/faq.html

9.6 JSSE Reference Guide

http://docs.oracle.com/javase/6/docs/technotes/guides/security/jsse/JSSERefGuide.html#SSLOverview

9.7 WCF – 2 Way SSL using Certificates

http://blogs.msdn.com/b/imayak/archive/2008/09/12/wcf-2-way-ssl-security-using-certificates.aspx

eMedNY neither endorses nor recommends any of the tools linked or referenced in this document. The intent here is strictly informational.

10 Web Service Definition Language (WSDL)

To assist in setting up your SOAP client, the WSDL for File Transfer Service follows. If it is more convenient, you can request an emailed copy from emednyproviderservices@csra.com :

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="http://org/emedny/fts/" xmlns:impl="http://org/emedny/fts/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:fts="http://org/emedny/fts/"
xmlns:intf="http://org/emedny/fts/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
  <wsdl:types>
    <xs:schema elementFormDefault="qualified" targetNamespace="http://org/emedny/fts/"
xmlns:tns="http://org/emedny/fts/" xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:import namespace="http://www.w3.org/2005/05/xmlmime"
schemaLocation="http://www.w3.org/2005/05/xmlmime"/>
    <xs:complexType name="Transaction">
      <xs:sequence>
        <xs:element maxOccurs="1" minOccurs="1" name="transData" type="xs:base64Binary"
xmime:expectedContentTypes="*/*"/>
        <xs:element maxOccurs="1" minOccurs="1" name="userName" type="xs:string"/>
        <xs:element maxOccurs="1" minOccurs="1" name="passWord" type="xs:string"/>
        <xs:element maxOccurs="1" minOccurs="1" name="transType" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="SendReceipt">
      <xs:sequence>
        <xs:element name="fileName" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>

    <xs:complexType name="FileData">
            <xs:sequence>
                    <xs:element name="fileData" type="xs:base64Binary" xmime:expectedContentTypes="*/*"/>
            </xs:sequence>
    </xs:complexType>

    <xs:complexType name="FileRequest">
       <xs:sequence>
        <xs:element maxOccurs="1" minOccurs="1" name="userName" type="xs:string"/>
        <xs:element maxOccurs="1" minOccurs="1" name="passWord" type="xs:string"/>
        <xs:element maxOccurs="1" minOccurs="1" name="fileName" type="xs:string"/>
       </xs:sequence>
    </xs:complexType>

    <xs:complexType name="FileNames">
            <xs:sequence>
                    <xs:element maxOccurs="unbounded" minOccurs="0" name="fileNames" type="xs:string"/>
            </xs:sequence>
            </xs:complexType>

    <xs:complexType name="FileNamesRequest">
            <xs:sequence>
                    <xs:element maxOccurs="1" minOccurs="1" name="userName" type="xs:string"/>
                    <xs:element maxOccurs="1" minOccurs="1" name="passWord" type="xs:string"/>
            </xs:sequence>
    </xs:complexType>

    <xs:element name="sendData">
      <xs:complexType>
        <xs:sequence>
          <xs:element maxOccurs="1" minOccurs="0" name="input" type="tns:Transaction"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>

    <xs:element name="sendDataResponse">
      <xs:complexType>
        <xs:sequence>
          <xs:element maxOccurs="1" minOccurs="0" name="output" type="tns:SendReceipt"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
```

```
  <xs:element name="getFile">
   <xs:complexType>
    <xs:sequence>
    <xs:element maxOccurs="1" minOccurs="0" name="input" type="tns:FileRequest"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>

  <xs:element name="getResponse">
   <xs:complexType>
    <xs:sequence>
    <xs:element maxOccurs="1" minOccurs="0" name="output" type="tns:FileData"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>

  <xs:element name="getFileNames">
   <xs:complexType>
    <xs:sequence>
            <xs:element maxOccurs="1" minOccurs="0" name="input" type="tns:FileNamesRequest"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>

  <xs:element name="getNamesResponse">
   <xs:complexType>
    <xs:sequence>
            <xs:element maxOccurs="1" minOccurs="0" name="output" type="tns:FileNames"/>
    </xs:sequence>
   </xs:complexType>
 </xs:element>

 <xs:element name="Fault">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="code" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
  </xs:element>

 </xs:schema>
</wsdl:types>
<wsdl:message name="getFileNamesRequest">
  <wsdl:part name="parameters" element="impl:getFileNames">
  </wsdl:part>
```

```
</wsdl:message>
<wsdl:message name="FTSFault">
 <wsdl:part name="parameters" element="impl:Fault">
 </wsdl:part>
</wsdl:message>
<wsdl:message name="getFileRequest">
 <wsdl:part name="parameters" element="impl:getFile">
 </wsdl:part>
</wsdl:message>
<wsdl:message name="sendDataRequest">
 <wsdl:part name="parameters" element="impl:sendData">
 </wsdl:part>
</wsdl:message>
<wsdl:message name="getFileResponse">
 <wsdl:part name="parameters" element="impl:getResponse">
 </wsdl:part>
</wsdl:message>
<wsdl:message name="sendDataResponse">
 <wsdl:part name="parameters" element="impl:sendDataResponse">
 </wsdl:part>
</wsdl:message>
<wsdl:message name="getFileNamesResponse">
 <wsdl:part name="parameters" element="impl:getNamesResponse">
 </wsdl:part>
</wsdl:message>
<wsdl:portType name="FTS">
 <wsdl:operation name="sendData">
  <wsdl:input name="sendDataRequest" message="impl:sendDataRequest">
 </wsdl:input>
  <wsdl:output name="sendDataResponse" message="impl:sendDataResponse">
 </wsdl:output>
  <wsdl:fault name="FTSFault" message="impl:FTSFault">
 </wsdl:fault>
 </wsdl:operation>
 <wsdl:operation name="getFile">
  <wsdl:input name="getFileRequest" message="impl:getFileRequest">
 </wsdl:input>
  <wsdl:output name="getFileResponse" message="impl:getFileResponse">
 </wsdl:output>
  <wsdl:fault name="FTSFault" message="impl:FTSFault">
 </wsdl:fault>
 </wsdl:operation>
 <wsdl:operation name="getFileNames">
  <wsdl:input name="getFileNamesRequest" message="impl:getFileNamesRequest">
 </wsdl:input>
  <wsdl:output name="getFileNamesResponse" message="impl:getFileNamesResponse">
 </wsdl:output>
  <wsdl:fault name="FTSFault" message="impl:FTSFault">
 </wsdl:fault>
 </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="FTS" type="impl:FTS">
 <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
 <wsdl:operation name="sendData">
  <soap:operation soapAction=""/>
  <wsdl:input name="sendDataRequest">
   <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="sendDataResponse">
   <soap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="FTSFault">
   <soap:fault name="FTSFault" use="literal"/>
  </wsdl:fault>
 </wsdl:operation>
 <wsdl:operation name="getFile">
  <soap:operation soapAction=""/>
  <wsdl:input name="getFileRequest">
   <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getFileResponse">
   <soap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="FTSFault">
   <soap:fault name="FTSFault" use="literal"/>
  </wsdl:fault>
 </wsdl:operation>
 <wsdl:operation name="getFileNames">
  <soap:operation soapAction=""/>
  <wsdl:input name="getFileNamesRequest">
   <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getFileNamesResponse">
   <soap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="FTSFault">
   <soap:fault name="FTSFault" use="literal"/>
  </wsdl:fault>
 </wsdl:operation>
</wsdl:binding>
<wsdl:service name="FTService">
 <wsdl:port name="FTSPort" binding="impl:FTS">
```

```
      <soap:address location="https://fts.emedny.org:8443/eMedNYServices/FTService/FTService"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

11 Change Log

Date	Ver	Modification
4/29/2013	1.1	Initial Version
5/15/2017	1.2	Minor revisions to the Introduction, added the WSDL to section 10
11/21/2019	1.3	Update Section 3 with system availability. Updated 3.2 with the limitations for getFileNames – it returns 500 responses Corrected names in Section 4 to be complaint with x12 standards. Added 834 files to 4.2 Update Section 9 with Contact information, and fixed links.
3/30/2019	1.31	Added 999 for 834 files to 4.1

eMedNY is the name of the electronic New York State Medicaid system. The eMedNY system allows New York Medicaid providers to submit claims and receive payments for Medicaid-covered services provided to eligible clients.

eMedNY offers several innovative technical and architectural features, facilitating the adjudication and payment of claims and providing extensive support and convenience for its users.

The information contained within this document was created in concert by eMedNY and DOH. More information about eMedNY can be found at www.emedny.org.