ACHWorks

API Guide Version 1.2

ACHWorks-SOAP Ver 4.0 API Guide

Table of Contents

1. Introduction

The use of web services like SOAP, both in theory and by experience , is said to be one of the most suitable applications for easy and seamless integration with existing or even new systems regardless of development platform and/or operating system. This is proven to be true for web services involving complex data structure and applications which requires frequent connections (data updates) like the ACH system. Since it's first release in 2001, ACHWorks-SOAP (Figure 1) has been a reliable and stable system.

The latest version of ACHWorks-SOAP (Version 4, 2010) provides new enhancements which include: (a) instant feedback of detailed status and/or connection errors for each method or operation call, (b) improved data structure for input and output eliminating the need for understanding the T$$ proprietary data format (as in Version 3), (c) changing from the XML-RPC to the Document/Literal format for compatibility with most and recent versions of development platforms, and (d) increase in the speed of connection.

The new version features a data structure that employs full object-oriented (OO) design that easily translates to non-fully OO formats like XML. Consequently, the generated classes from the web service WSDL present data as objects. This feature facilitates easier and more intuitive consuming of the web service i.e. creation of a SOAP client application.

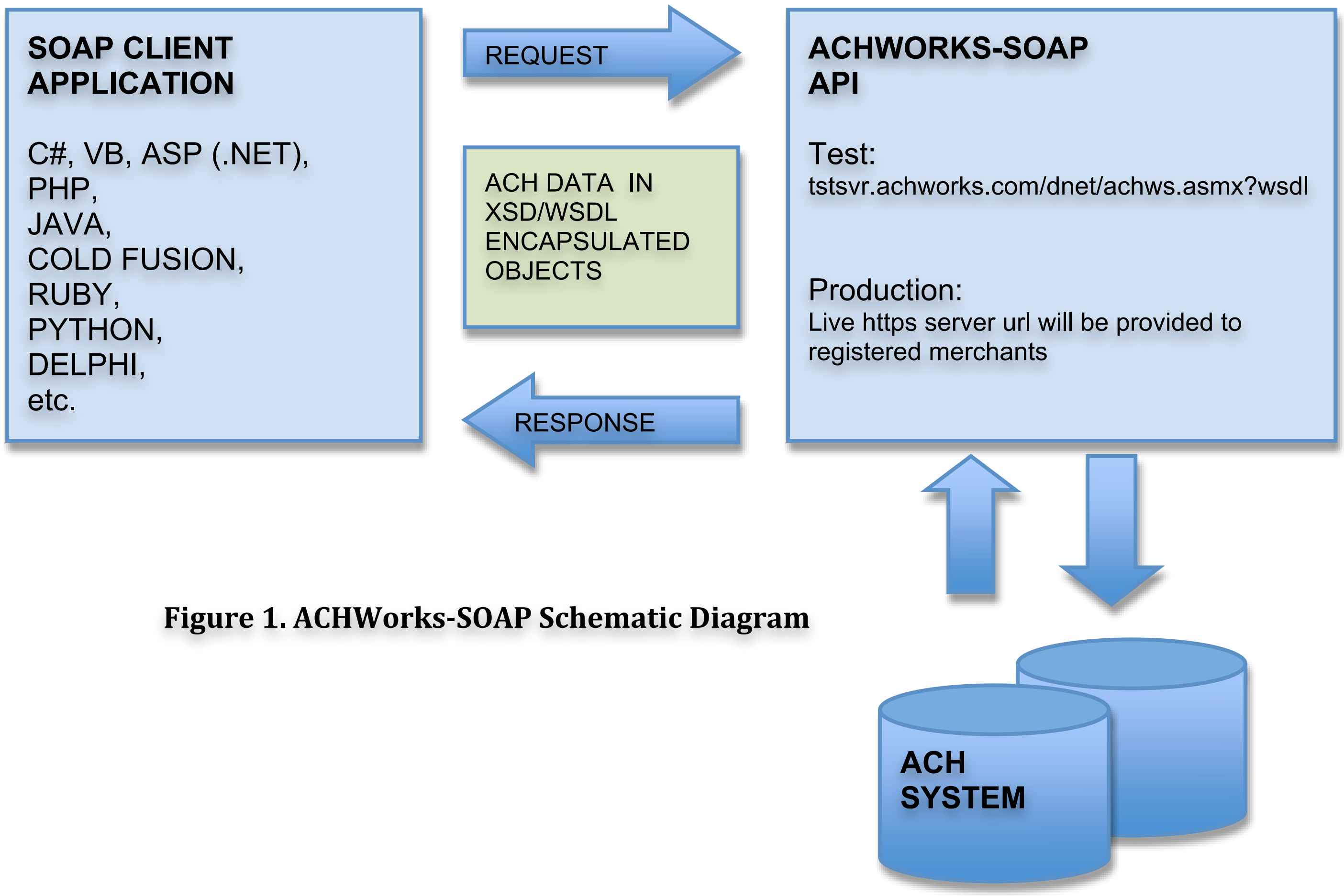

Figure 1. ACHWorks-SOAP Schematic Diagram

2. Description of Methods/Operations

This section lists the methods and description for using the ACHWorks-SOAP web service. A brief description is provided for each method to complement the service description (test WSDL url) which can be found on this link: http://tstsvr.achworks.com/dnet/achws.asmx?WSDL . A different link for WSDL to send live ACH transactions is provided for registered merchants.

For details of the data structure and definitions, please refer to Section 3 of this guide.

Viewing the sample codes, Appendix A2 in this guide, also prove useful in understanding on how a method is called in an application and to easily get started quickly.

2.1. Checking Connection and Account Status

2.1.1 ConnectionCheck

This is the basic method for accessing the web service to determine validity of company credentials. This method may be used for initial testing of the service only. To view the Request/Response description of this operation go to this link: http://tstsvr.achworks.com/dnet/achws.asmx?op=ConnectionCheck.

Format:
ConnectionCheckResult = ***ConnectionCheck***(InpCompanyInfo),
    where ConnectionCheckResult is of type String
        InpCompanyInfo is of type CompanyInfo (Section 3.1)

Example:
The InpCompanyInfo can have the following value,

```
InpCompanyInfo.SSS=”TST”
InpCompanyInfo.LocID=”9505”
InpCompanyInfo.Company=”MYCOMPANY”
InpCompanyInfo.CompanyKey=”SASD%!%$DGLJGWYRRDGDDUDFDESDHDD”
```

and the ConnectionCheckResult object may have the following value after the method call,

```
ConnectionCheckResult -> “SUCCESS:Valid Account”
```

or if rejected,

```
ConnectionCheckResult -> “REJECTED:Invalid Account”
```

2.1.2 CheckCompanyStatus

This method checks the status of the company credentials whether it’s an active or an inactive account. It is possible that an account is valid (e.g. ConnectionCheck) but not yet activated or already been deactivated or closed. An account must be active to initiate methods for sending ACH transactions and getting ACH returns. To view the Request/Response description of this operation go to this link: http://tstsvr.achworks.com/dnet/achws.asmx?op=CheckCompanyStatus.

Format:
CheckCompanyStatusResult = ***CheckCompanyStatus***(InpCompanyInfo),
    where CheckCompanyStatusResult is of type String
        InpCompanyInfo is of type CompanyInfo (Section 3.1)

Example:
The InpCompanyInfo can have the following value,

```
InpCompanyInfo.SSS="TST"
InpCompanyInfo.LocID="9505"
InpCompanyInfo.Company="MYCOMPANY"
InpCompanyInfo.CompanyKey="SASD%!%$DGLJGWYRRDGDDUDFDESDHDD"
```

and the CheckCompanyStatusResult object may have the following value after the method call,

```
CheckCompanyStatusResult -> "SUCCESS:Valid And Active Account"
```

or if rejected,

```
CheckCompanyStatusResult -> "REJECTED:Invalid And/Or Inactive Account"
```

2.2. Sending Transactions
2.2.1 SendACHTrans

This method is used for sending a single ACH transaction or record. ACHTransRecord describes the format of a single ACH transaction or record (see Section 3.2). To view the Request/Response description of this operation go to this link: http://tstsvr.achworks.com/dnet/achws.asmx?op=SendACHTrans

Format:
SendACHTransResult = ***SendACHTrans***(InpCompanyInfo, InpACHTransRecord),
where SendACHTransResult is of type TransResult (Section 3.6)
InpCompanyInfo is of type CompanyInfo (Section 3.1)
InpACHTransRecord is of type ACHTransRecord (Section 3.2)

Example:
The input InpCompanyInfo and InpACHTransRecord objects may have the following values,

```
InpCompanyInfo.SSS="TST"
InpCompanyInfo.LocID="9502"
InpCompanyInfo.Company="MYCOMPANY"
InpCompanyInfo.CompanyKey="SASD%!%$DGLJGWYRRDGDDUDFDESDHDD"

InpACHTransRecord.SSS="TST"
InpACHTransRecord.LocID="9502"
InpACHTransRecord.FrontEndTrace="XC#-0002"
InpACHTransRecord.CustomerName="DOE, JOHN"
InpACHTransRecord.CustomerRoutingNo="123456780"
InpACHTransRecord.CustomerAcctNo="00002323044"
.      .      .      .      .      .      .
.      .      .      .      .      .      .
InpACHTransRecord.AccountSet="1"
```

Then, the SendACHTransResult object may have the following value after the method call,

```
SendACHTransResult.SSS->"TST"
SendACHTransResult.LocID->"9502"
SendACHTransResult.Status->"SUCCESS"
SendACHTransResult.Details->"Transaction record received and queued on 04/16/2010 03:34:34 PM Pacific Time
                            [FrontEndTrace=X9000123310]. Server Processing Time:15.625 ms"
SendACHTransResult.TotalNumErrors ->0
```

Or if rejected,

```
SendACHTransResult.SSS->"TST"
SendACHTransResult.LocID->"9502"
SendACHTransResult.Status->"REJECTED"
SendACHTransResult.Details->"Rejected due to errors"
SendACHTransResult.TotalNumErrors ->2
SendACHTransResult.Errors[0]->"Error#1:FrontEndTrace X9000123310 already used."
SendACHTransResult.Errors[1]->"Error#2:Unable to complete processing of data due to error(s)."
```

Special Note: All transactions sent via this method can be separately managed using the ACHWorks-WEB online application (optional but useful feature for online viewing and editing of transactions). Although

ACHWorks-WEB requires a separate license, the license is not required to facilitate the processing of the transaction sent using this method.

2.2.2 SendACHTransBatch

This method is used for sending multiple or batch of ACH transactions. ACHFile describes the format of a batch file (see Section 3.3). To view the Request/Response description of this operation go to this link: http://tstsvr.achworks.com/dnet/achws.asmx?op=SendACHTransBatch

Format:
SendACHTransBatchResult = ***SendACHTransBatch***(InpCompanyInfo, InpACHFile),
where SendACHTransBatchResult is of type TransResult (Section 3.6)
InpCompanyInfo is of type CompanyInfo (Section 3.1)
InpACHFile is of type ACHFile (Section 3.3)

Example:
The input InpCompanyInfo and InpACHFile objects may have the following values,

```
  InpCompanyInfo.SSS=”TST”
  InpCompanyInfo.LocID=”9502”
  InpCompanyInfo.Company=”MYCOMPANY”
  InpCompanyInfo.CompanyKey=”SASD%!%$DGLJGWYRRDGDDUDFDESDHDD”

  InpACHFile.SSS=”TST”
  InpACHFile.LocID=”9502”
  InpACHFile.ACHFileName=””  //we recommend that you leave this blank,  see Section 3.3.3 for forma
  InpACHFile.TotalNumRecords=3
  InpACHFile.TotalDebitRecords=2
  InpACHFile.TotalDebitAmount=600.5
  InpACHFile.TotalCreditRecords=1
  InpACHFile.TotalCreditAmount=350.25
  InpACHFile.ACHRecords[0]=InpACHTransRecord1  //See example in Section 2.2.2 for assigning InpACHransRecord
  InpACHFile.ACHRecords[1]=InpACHTransRecord2  //See example in Section 2.2.2 for assigning InpACHransRecord
 InpACHFile.ACHRecords[2]=InpACHTransRecord3   //See example in Section 2.2.2 for assigning InpACHransRecord
```

Then, the SendACHTransBatchResult object may have the following value after the method call,

```
  SendACHTransBatchResult.SSS->”TST”
  SendACHTransBatchResult.LocID->”9502”
  SendACHTransBatchResult.Status->”SUCCESS”
  SendACHTransBatchResult.Details->”Transaction records received and queued on 04/16/2010 04:09:25 PM Pacific Time
                                   [ACHFileName=TST9502-10041600.XML]. Server Processing Time:<0.0001 ms”
  SendACHTransBatchResult TotalNumErrors ->0
```

Or if rejected,

```
  SendACHTransBatchResult.SSS->”TST”
  SendACHTransBatchResult.LocID->”9502”
  SendACHTransBatchResult.Status->”REJECTED”
  SendACHTransBatchResult.Details->”Rejected due to errors”
  SendACHTransBatchResult.TotalNumErrors ->2
  SendACHTransBatchResult.Errors[0]->”Error#1: Record count defined does not match actual number of records.”
```

```
SendACHTransBatchResult.Errors[1]->"Error#2:Unable to complete processing of data due to error(s)."
```

2.2.3 SendACHTransBatchBin

Some developers prefer to send a ready-made file instead of assembling it in their client code. This method can be used for sending multiple or batch of ACH transactions as an alternative to the SendACHTransBatch method. The input ACHFile is encoded in a binary format (Figure 2.1). This method is also recommended if sending more than 5,000 transactions per send call. To view the Request/Response description of this operation go to this link: http://tstsvr.achworks.com/dnet/achws.asmx?op=SendACHTransBatchBin

Format:
SendACHTransBatchBinResult = ***SendACHTransBatchBin***(InpCompanyInfo, InpBinFile),
where SendACHTransBatchBINResult is of type TransResult (Section 3.6)
InpCompanyInfo is of type CompanyInfo (Section 3.1)
InpBinFile is of type base64binary, binary encoded ACHFile (see Figure 2.1 below)

Example:
Then input InpCompanyInfo and InpACHFile objects may have the following values,

```
InpCompanyInfo.SSS="TST"
InpCompanyInfo.LocID="9502"
InpCompanyInfo.Company="MYCOMPANY"
InpCompanyInfo.CompanyKey="SASD%!%$DGLJGWYRRDGDDUDFDESDHDD"

InpBinFile = //assign a base64 binary encoded of the ACHFile, see Figure 2.1 below
```

Then, the SendACHTransBatchBINResult object may have the following value after the method call,

```
SendACHTransBatchResult.SSS->"TST"
SendACHTransBatchResult.LocID->"9502"
SendACHTransBatchResult.Status->"SUCCESS"
SendACHTransBatchResult.Details->"Transaction records received and queued on 04/16/2010 04:09:25 PM Pacific Time
                                  [ACHFileName=TST9502-10041600.XML]. Server Processing Time:<0.0001 ms"
SendACHTransBatchResult TotalNumErrors ->0
```

Or if rejected,

```
SendACHTransBatchResult.SSS->"TST"
SendACHTransBatchResult.LocID->"9502"
SendACHTransBatchResult.Status->"REJECTED"
SendACHTransBatchResult.Details->"Rejected due to errors"
SendACHTransBatchResult.TotalNumErrors ->2
SendACHTransBatchResult.Errors[0]->"Error#1: Record count defined does not match actual number of records."
SendACHTransBatchResult.Errors[1]->"Error#2:Unable to complete processing of data due to error(s)."
```

Figure 2.1 Sample ACHFile text (to be encoded in base64 binary format as InpBinFile):

```
 <?xml version="1.0" encoding="utf-8" ?>
<ACHFile>
        <SSS>TST</SSS>
        <LocID>9505</LocID>
        <ACHFileName>TST9505-10040102.XML</ACHFileName>
```

```
<TotalNumRecords>3</TotalNumRecords>
<TotalDebitRecords>2</TotalDebitRecords>
<TotalDebitAmount>600.5</TotalDebitAmount>
<TotalCreditRecords>1</TotalCreditRecords>
<TotalCreditAmount>350.25</TotalCreditAmount>
<ACHRecords>
        <ACHTransRecord>
                <SSS>TST</SSS>
                <LocID>9505</LocID>
                <FrontEndTrace>J-00081</FrontEndTrace>
                <OriginatorName>MYCOMPANY</ OriginatorName >
                <TransactionCode>PPD</TransactionCode>
                <CustTransType>D</CustTransType>
                <CustomerID>CUSTID123-1</CustomerID>
                <CustomerName>DOE, JOHN</CustomerName>
                <CustomerRoutingNo>987654320</CustomerRoutingNo>
                <CustomerAcctNo>00332358882</CustomerAcctNo>
                <CustomerAcctType>C</CustomerAcctType>
                <TransAmount>100.25</TransAmount>
                <CheckOrCustID>9166388811</CheckOrCustID>
                <CheckOrTransDate>03/24/2010</CheckOrTransDate>
                <EffectiveDate>03/24/2010</EffectiveDate>
                <Memo>FirstPay</Memo>
                <OpCode>S</OpCode>
                <AccountSet>1</AccountSet>
        </ACHTransRecord>
        <ACHTransRecord>
                <SSS>TST</SSS>
                <LocID>9505</LocID>
                <FrontEndTrace>J-00082</FrontEndTrace>
                < OriginatorName >MYCOMPANY</ OriginatorName >
                <TransactionCode>PPD</TransactionCode>
                <CustTransType>D</CustTransType>
                <CustomerID>CUSTID123-2</CustomerID>
                <CustomerName>SMITH, JANET</CustomerName>
                <CustomerRoutingNo>987654320</CustomerRoutingNo>
                <CustomerAcctNo>00332358882</CustomerAcctNo>
                <CustomerAcctType>C</CustomerAcctType>
                <TransAmount>500.25</TransAmount>
                <CheckOrCustID>9166388811</CheckOrCustID>
                <CheckOrTransDate>03/24/2010</CheckOrTransDate>
                <EffectiveDate>03/24/2010</EffectiveDate>
                <Memo>FirstPay</Memo>
                <OpCode>S</OpCode>
                <AccountSet>1</AccountSet>
        </ACHTransRecord>
        <ACHTransRecord>
                <SSS>TST</SSS>
                <LocID>9505</LocID>
                <FrontEndTrace>J-00083</FrontEndTrace>
                < OriginatorName >MYCOMPANY</ OriginatorName >
                <TransactionCode>PPD</TransactionCode>
                <CustTransType>C</CustTransType>
                <CustomerID>CUSTID123-3</CustomerID>
                <CustomerName>YOUNG, JOE</CustomerName>
```

```
                    <CustomerRoutingNo>987654320</CustomerRoutingNo>
                    <CustomerAcctNo>00332358882</CustomerAcctNo>
                    <CustomerAcctType>C</CustomerAcctType>
                    <TransAmount>350.25</TransAmount>
                    <CheckOrCustID>9166388811</CheckOrCustID>
                    <CheckOrTransDate>03/24/2010</CheckOrTransDate>
                    <EffectiveDate>03/24/2010</EffectiveDate>
                    <Memo>FirstPay</Memo>
                    <OpCode>S</OpCode>
                    <AccountSet>1</AccountSet>
               </ACHTransRecord>
          </ACHRecords>
</ACHFile>
```

2.3 Getting Settlements and Returns
2.3.1 GetACHReturns

This method is used to retrieve outstanding status of Returns and Settlements from the ACH system for a transaction or set of transactions. The output is stored in ACHReturns file described in Section 3.5. A file may consist of one or more list or array of return records. Each return record is formatted as an ACHReturnRecord (Section 3.4).

This is the recommended way to check for status of transactions. Only new statuses (not previously picked up) will be available using this method. To view the Request/Response description of this operation go to this link: http://tstsvr.achworks.com/dnet/achws.asmx?op=GetACHReturns

Format:
GetACHReturnsResult = ***GetACHReturns***(InpCompanyInfo),
where GetACHReturnsResult is of type ACHReturns (Section 3.5)
InpCompanyInfo is of type CompanyInfo (Section 3.1)

Example:
The InpCompanyInfo can have the following value,

```
InpCompanyInfo.SSS=”TST”
InpCompanyInfo.LocID=”9505”
InpCompanyInfo.Company=”MYCOMPANY”
InpCompanyInfo.CompanyKey=”SASD%!%$DGLJGWYRRDGDDUDFDESDHDD”
```

and the GetACHReturnResult object may have the following value after the method call,

```
GetACHReturnResult.SSS->”TST”
GetACHReturnResult.LocID->”9502”
GetACHReturnResult.Status->”SUCCESS”
GetACHReturnResult.Details -> “Retrieved 14 records on 04/16/2010 04:49:27 PM Pacific Time.
                               Server Processing Time:<0.0001 ms”
GetACHReturnResult.TotalNumRecords -> 14
GetACHReturnResult.ACHReturnRecords[0].FrontEndTrace -> “X35948”
GetACHReturnResult.ACHReturnRecords[0].ResponseCode -> “1SNT”
GetACHReturnResult.ACHReturnRecords[0].TransAmount -> 356.45
GetACHReturnResult.ACHReturnRecords[0].EffectiveDate -> “2010-03-05T00:00:00”
GetACHReturnResult.ACHReturnRecords[1].FrontEndTrace -> “X35950”
GetACHReturnResult.ACHReturnRecords[1].ResponseCode -> “1SNT”
GetACHReturnResult.ACHReturnRecords[1].TransAmount -> 50.25
GetACHReturnResult.ACHReturnRecords[1].EffectiveDate -> “2010-03-05T00:00:00”
GetACHReturnResult.ACHReturnRecords[2].FrontEndTrace -> “X35951”
GetACHReturnResult.ACHReturnRecords[2].ResponseCode -> “2STL”
GetACHReturnResult.ACHReturnRecords[2].TransAmount -> 750.25
GetACHReturnResult.ACHReturnRecords[2].EffectiveDate -> “2010-03-05T00:00:00”
GetACHReturnResult.ACHReturnRecords[2].ActionDetail -> “A340016R Credit 01 TS 00001”
.                                           .     .      .      .
.                                           .     .      .      .
GetACHReturnResult.ACHReturnRecords[14].FrontEndTrace -> “X78531”
.                                           .     .      .      .
.                                           .     .      .      .
```

```
  GetACHReturnResult.ACHReturnRecords[14].ActionDetail -> "A340016W Credit 03 TS 00003"
```

Or if no data to retrieve,

```
  GetACHReturnResult.SSS->"TST"
  GetACHReturnResult.LocID->"9502"
  GetACHReturnResult.Status->"SUCCESS"
  GetACHReturnResult.Details -> "No records to retrieve at this time, 04/20/2010 04:58:26 PM Pacific Time.
                                 Server Processing Time:<0.0001 ms"
```

2.3.2 GetACHReturnsHist

This method is used to retrieve status at specified date range of Returns and Settlements from the ACH system for a transaction or set of transactions. The output is stored in ACHReturns file described in Section 3.5. A file may consist of one or more list or array of return records. Each return record is formatted as an ACHReturnRecord (Section 3.4). This method is useful for rebuilding historical data in the event of hard drive failure or data corruption. To view the Request/Response description of this operation go to this link: http://tstsvr.achworks.com/dnet/achws.asmx?op=GetACHReturnsHist

Format:

GetACHReturnsHistResult = ***GetACHReturnsHist***(InpCompanyInfo,ReturnDateFrom,ReturnDateTo),
  where GetACHReturnsHistResult is of type ACHReturns (Section 3.5)
    InpCompanyInfo is of type CompanyInfo (Section 3.1)
    ReturnDateFrom and ReturnDateTo are of type dateTime (please refer to sample codes for correct implementation of dateTime depending on the development platform)

Example:

The InpCompanyInfo can have the following value,

```
  InpCompanyInfo.SSS="TST"
  InpCompanyInfo.LocID="9505"
  InpCompanyInfo.Company="MYCOMPANY"
  InpCompanyInfo.CompanyKey="SASD%!%$DGLJGWYRRDGDDUDFDESDHDD"
```

and the GetACHReturnHistResult object may have the following value after the method call,

```
  GetACHReturnResultHist.SSS->"TST"
  GetACHReturnResultHist.LocID->"9502"
  GetACHReturnResultHist.Status->"SUCCESS"
  GetACHReturnResultHist.Details -> "Retrieved 142 records on 04/16/2010 04:49:27 PM Pacific Time.
                                     Server Processing Time:<0.0001 ms"
  GetACHReturnResultHist.TotalNumRecords -> 142
  GetACHReturnResultHist.ReturnDateFrom –>"2010-03-25T00:00:00"
  GetACHReturnResultHist.ReturnDateTo –>"2010-03-29T00:00:00"
  GetACHReturnResultHist.ACHReturnRecords[0].FrontEndTrace -> "X35948"
  GetACHReturnResultHist.ACHReturnRecords[0].ResponseCode -> "1SNT"
  GetACHReturnResultHist.ACHReturnRecords[0].TransAmount -> 356.45
  GetACHReturnResultHist.ACHReturnRecords[0].EffectiveDate -> "2010-03-05T00:00:00"
  GetACHReturnResultHist.ACHReturnRecords[1].FrontEndTrace -> "X35950"
  GetACHReturnResultHist.ACHReturnRecords[1].ResponseCode -> "1SNT"
  GetACHReturnResultHist.ACHReturnRecords[1].TransAmount -> 50.25
  GetACHReturnResultHist.ACHReturnRecords[1].EffectiveDate -> "2010-03-05T00:00:00"
  GetACHReturnResultHist.ACHReturnRecords[2].FrontEndTrace -> "X35951"
  GetACHReturnResultHist.ACHReturnRecords[2].ResponseCode -> "2STL"
  GetACHReturnResultHist.ACHReturnRecords[2].TransAmount -> 750.25
  GetACHReturnResultHist.ACHReturnRecords[2].EffectiveDate -> "2010-03-05T00:00:00"
  GetACHReturnResultHist.ACHReturnRecords[2].ActionDetail -> "A340016R Credit 01 TS 00001"
  .                                           .     .       .       .
  .                                           .     .       .       .
  GetACHReturnResultHist.ACHReturnRecords[142].FrontEndTrace -> "X78531"
  .                                           .     .       .       .
  .                                           .     .       .       .
  GetACHReturnResultHist.ACHReturnRecords[142].ActionDetail -> "A340016W Credit 03 TS 00003"
```

Or if no data to retrieve,

```
  GetACHReturnHistResult.SSS->"TST"
  GetACHReturnHistResult.LocID->"9502"
  GetACHReturnHistResult.Status->"SUCCESS"
 GetACHReturnHistResult.Details -> "No records to retrieve at this time, 04/20/2010 04:58:26 PM Pacific Time.
                                   Server Processing Time:<0.0001 ms"
```

A derivate method called **GetACHReturnsHistRC** can be used to retrieve ACH Settlements and Returns by response code. Response codes is either a 1SNT, 2STL, 3RET, 4INT, 5COR or 9BNK (see Section 3.4.5).

Format:

GetACHReturnsHistRCResult = ***GetACHReturnsHistRC***(InpCompanyInfo,ReturnDateFrom,ReturnDateTo, InpResponseCode),

where GetACHReturnsHistRCResult is of type ACHReturns (Section 3.5)

InpCompanyInfo is of type CompanyInfo (Section 3.1)

ReturnDateFrom and ReturnDateTo are of type dateTime (please refer to sample codes for correct implementation of dateTime depending on the development platform)

InpResponseCode is of type String (Section 3.4.5)

2.4. Retrieving Past Connection Results and Errors

2.4.1 GetResultFile

This method is used to retrieve past results of connections. The output is stored in ResultFile described in Section 3.7. A file may consist of one or more list or array of result records. Each result record is formatted as a TransResult (Section 3.6). To view the Request/Response description of this operation go to this link: http://tstsvr.achworks.com/dnet/achws.asmx?op=GetResultFile

Format:
GetResultFileResult = ***GetResultFile***(InpCompanyInfo,ResultDateFrom,ResultDateTo),
where GetResultFileResult is of type ResultFile (Section 3.7)
InpCompanyInfo is of type CompanyInfo (Section 3.1)
ResultDateFrom and ResultDateTo are of type dateTime (please refer to sample codes for correct implementation of dateTime depending on the development platform)

Example:
Calling GetResultFile may have following values for the GetResultFileResult object,

```
GetResultFileResult.SSS ->"TST"
GetResultFileResult.LocID ->"9505"
GetResultFileResult.Status ->"SUCCESS"
GetResultFileResult.Details->" Retrieved 15 connection records on 04/16/2010 02:53:16 PM Pacific Time ...."
GetResultFileResult.ResultDateFrom –>"2010-03-25T00:00:00"
GetResultFileResult.ResultDateTo –>"2010-03-29T00:00:00"
GetResultFileResult.TotalResultRecords ->15
GetResultFileResult.TransResults[0].CallMethod -> "SendACHTransBatch"
GetResultFileResult.TransResults[0].Status -> "SUCCESS"
GetResultFileResult.TransResults[0].FileName -> "TST9505-1003251B.XML"
GetResultFileResult.TransResults[1].CallMethod -> "SendACHTransBatch"
GetResultFileResult.TransResults[1].Status -> "SUCCESS"
GetResultFileResult.TransResults[1].FileName -> "TST9505-1003251C.XML"
GetResultFileResult.TransResults[2].CallMethod -> "SendACHTrans"
GetResultFileResult.TransResults[2].Status -> "SUCCESS"
GetResultFileResult.TransResults[2].FrontEndTrace -> "X9000123310454"
.          .               .          .          .   .  .    .    .
.          .               .          .          .   .  .    .    .
GetResultFileResult.TransResults[15].CallMethod -> "SendACHTransBatch"
GetResultFileResult.TransResults[15].Status -> "REJECTED"
GetResultFileResult.TransResults[15].FileName -> "TST9505-10032515.XML"
```

2.4.2 GetErrorFile

This method is used to retrieve past errors in connections. The output is stored in ErrorFile described in Section 3.9. A file may consist of one or more list or array of error records. Each error record is formatted as an ErrorRecord (Section 3.8). To view the Request/Response description of this operation go to this link: http://tstsvr.achworks.com/dnet/achws.asmx?op=GetErrorFile

Format:
GetErrorFileResult = ***GetErrorFile***(InpCompanyInfo,ErrorDateFrom,ErrorDateTo),
where GetErrorFileResult is of type ErrorFile (Section 3.9)
InpCompanyInfo is of type CompanyInfo (Section 3.1)
ErrorDateFrom and ErrorDateTo are of type dateTime (please refer to sample codes for correct implementation of dateTime depending on the development platform)

Example:
Calling GetErrorFile may have following values for the GetErrorFileResult object,

```
GetErrorFileResult.SSS ->"TST"
GetErrorFileResult.LocID ->"9505"
GetErrorFileResult.Status ->"SUCCESS"
GetErrorFileResult.Details->"Retrieved 9 error records on 04/16/2010 02:09:39 PM Pacific Time ...."
GetErrorFileResult.ErrorDateFrom –>"2010-03-25T00:00:00"
GetErrorFileResult.ErrorDateTo –>"2010-03-29T00:00:00"
GetErrorFileResult.TotalErrorRecords ->9
GetErrorFileResult.ErrorRecords[0].CallMethod -> "SendACHTrans"
GetErrorFileResult.ErrorRecords[0].FrontEndTrace -> "XT-23201"
GetErrorFileResult.ErrorRecords[0].TotalNumErrors -> 2
GetErrorFileResult.ErrorRecords[0].Errors[0] -> "Error#1:CustomerRoutingNo is a required field."
GetErrorFileResult.ErrorRecords[0].Errors[1] -> "Error#2:Unable to complete processing of data due to error(s)."
GetErrorFileResult.ErrorRecords[1].CallMethod -> "SendACHTransBatch"
GetErrorFileResult.ErrorRecords[1].FileName -> "TST9505-10032512.XML"
GetErrorFileResult.ErrorRecords[1].TotalNumErrors -> 2
GetErrorFileResult.ErrorRecords[1].Errors[0] -> "Error#1: Invalid CustomerAcctType, Rec#1."
GetErrorFileResult.ErrorRecords[1].Errors[1] -> "Error#2:Unable to complete processing of data due to error(s)."
.          .              .         .         .  .  .    .    .
.          .              .         .         .  .  .    .    .
GetErrorFileResult.ErrorRecords[9].CallMethod -> "SendACHTransBatch"
GetErrorFileResult.ErrorRecords[9].FileName -> "TST9505-1003251A.XML"
GetErrorFileResult.ErrorRecords[9].TotalNumErrors -> 3
GetErrorFileResult.ErrorRecords[9].Errors[0] -> "Error#1: Invalid CustomerAcctType, Rec#1."
GetErrorFileResult.ErrorRecords[9].Errors[1] -> "Error#2: TransAmount has invalid or zero value, Rec#1."
GetErrorFileResult.ErrorRecords[9].Errors[2] -> "Error#3:Unable to complete processing of data due to error(s)."
```

2.5. Viewing and Deleting Batch Transaction

2.5.1 ViewACHTransBatch

This method is used to view or read an ACH Batch File specified by the ACHFileName. The output is stored in an ACHFile described in Section 3.3. A file may consist of one or more list or array of ACHTransRecords. Each transaction record is formatted as an ACHTransRecord (Section 3.2). To view the Request/Response description, go to this link: http://tstsvr.achworks.com/dnet/achws.asmx?op=ViewACHTransBatch

Format:
ViewACHTransBatchResult = ***ViewACHTransBatch***(InpCompanyInfo, InpACHFileName),
where ViewACHTransBatchResult is of type ACHFile (Section 3.3)
InpCompanyInfo is of type CompanyInfo (Section 3.1)
InpACHFileName is of type String

2.5.2 DeleteACHTransBatch

This method is used to delete an ACH Batch File specified by the ACHFileName (if it still possible i.e. file not sent yet to ACH for processing or not previously deleted).

Format:
DeleteACHTransBatchResult = ***DeleteACHTransBatch***(InpCompanyInfo, InpACHFileName),
where DeleteACHTransBatchResult is of type TransResult (Section 3.6)
InpCompanyInfo is of type CompanyInfo (Section 3.1)
InpACHFileName is of type String

To view the Request/Response description of this operation go to this link:
http://tstsvr.achworks.com/dnet/achws.asmx?op=DeleteACHTransBatch

Special Note: Only files or transactions sent via the SendACHTransBatch and SendACHTransBatchBIN methods can be deleted using this method. Deleting a single transaction (sent using SendACHTrans) can be separately done using the ACHWorks-WEB online application, an optional but useful feature for online viewing and editing of transactions. ACHWorks-WEB requires a separate license.

3. Data Definitions

This lists the data types for input/output in using the ACHWorks-SOAP web service. A brief description is provided for each method to complement the service description (WSDL url) which can be found on this link: http://tstsvr.achworks.com/dnet/achws.asmx?WSDL .

Please see Section 3.10 for required field types and lengths.

3.1 CompanyInfo

The CompanyInfo serves as the basic company credential for accessing the ACHWorks-SOAP web service. This is a required input for each method or operation. It consists of SSS, LocID, Company and CompanyKey which will be described below.

3.1.1 SSS

SSS is a 3-letter code furnished to registered merchants. For sample codes and for using the test server, the value 'TST' is commonly used.

3.1.2 LocID

LocID is a 4-digit alphanumeric code furnished to registered merchants. For sample codes and for using the test server, the value '9502' or '9505' is commonly used.

3.1.3 Company

Company is an alpha-numeric user ID (used as username for the service) furnished to registered merchants. For sample codes and for using the test server, the value 'MY COMPANY' or 'THAT COMPANY' is commonly used.

3.1.4 CompanyKey

CompanyKey is an alpha-numeric user KEY (used as password for the service) furnished to registered merchants.

3.2 ACHTransRecord

The ACHTransRecord represents the basic or single ACH record or transaction. It is used as input for sending an ACH transaction i.e. using the SendACHTrans method. It consists of SSS, LocID, FrontEndTrace, OriginatorName, TransactionCode, CustTransType, CustomerID, CustomerName, CustomerRoutingNo, CustomerAcctNo, TransAmount, CheckOrCustID, CheckOrTransDate, EffectiveDate, Memo, OpCode, and AccountSet which will be described below. For details on the field types and format please refer to Table 3.1.

3.2.1 SSS
SSS is a 3-letter code furnished to registered merchants.   For sample codes and for using the test server, the value 'TST' is commonly used.

3.2.2 LocID
LocID is a 4-digit alphanumeric code furnished to registered merchants. For sample codes and for using the test server, the value '9502' or '9505' is commonly used.

3.2.3 FrontEndTrace
The FrontEndTrace is used to track this particular record or transaction. A unique value should be used for this field. This field will always be returned for referencing this record. Important Note: The FrontEndTrace number should not start with the character W to ensure uniqueness especially when using the SendACHTrans method (Transactions entered manually through ACHWorks-WEB automatically be assigned a FrontEndTrace beginning with the char "W").

3.2.4 OriginatorName
The OriginatorName is the name of the company sending the transaction. This is typically the name of the company that owns the ACH account (or it's abbreviation). This name will appear on th receiver (customer's) bank statement. It is important that the OriginatorName or Abbreviation is recognizable by the receiver to reduce the possibility of a chargeback for unauthorized transaction. For RCK transactions, this is the original payee in the Check i.e. the name written on the 'Pay to the order of' line.

3.2.5 TransactionCode
The TransactionCode is the ACH transaction codes such as PPD, CCD, TEL, WEB, RCK, POS, POP and ARC. The code indicates what type of authorization has been obtained to approve the transaction. The most common codes are PPD which is a signed authorization for a Debit or Credit to a Personal bank account and CCD which is a signed authorization do Debit or Credit a business bank account. WEB is used for transactions authorizations obtained from consumers through the internet with an e-signature. TEL is for consumer authorizations obtained over the telephone. Please contact T$$/ACHWorks for assistance in classifying transactions codes to be used.

3.2.6 CustTransType
CustTransType is the transaction type. It indicates if it is a debit or a credit to the receiver's account, 'D' for debit and 'C' for credit.

3.2.7 CustomerID
CustomerID is an optional field for assigning a unique identifier for a customer.

3.2.8 CustomerName
CustomerName is the name of the customer. Preferred format is Lastname, Firstname and all caps. The value is trimmed to the maximum length allowed by the ACH which 22 characters.

3.2.9 CustomerRoutingNo
This is the customer's 9-digit Bank Routing Number.

3.2.10 CustomerAcctNo
This is the customer's Bank Account Number.

3.2.11 CustomerAcctType
This is the customer's Bank Account Type either 'C' for checking account or 'S' for savings account.

3.2.12 TransAmount
TransAmount is the amount of the transaction in US dollars, e.g. 575.45

3.2.13 CheckOrCustID
CheckOrCustID is the check number. This field may contain the CustomerID if no value is provided.

3.2.14 CheckOrTransDate
CheckOrTransDate is the original date printed on the Check or usually the date of the transaction.

3.2.15 EffectiveDate
EffectiveDate is the date that you want this transaction submitted to the ACH system. If date is ahead of current date, the transaction will be warehoused in our server.

3.2.16 Memo
The memo is the ten character field which will appear on the customer's bank statement.

3.2.17 OpCode
OpCode is an optional code which is either 'S' for single entry (default) or 'R' for recurring entry. This code is required for POS and WEB TransactionCodes.

3.2.18 AccountSet

This value is furnished for registered merchants. A merchant may have multiple Account Set numbers for each of the merchant's bank accounts registered with T$$, a corresponding Account Set number will be defined in the server. I.E. Account set 1 is your B of A account, Account Set 2 is your Wells Fargo account, etc.

3.3 ACHFile

The ACHFile represents the file with elements or fields that describe an ACH file. It contains the batch of transactions in the form ACHRecords (same as the list or array of ACHTransRecord's). It is used as input for sending batch of ACH transactions i.e. using the SendACHTransBatch method. It consists of SSS, LocID, ACHFileName, TotalNumRecords, TotalDebitRecords, TotalCreditRecords, TotalDebitAmount, TotalCreditAmount and the ACHRecords which will be described below.

3.3.1 SSS

SSS is a 3-letter code furnished to registered merchants. For sample codes and for using the test server, the value 'TST' is commonly used.

3.3.2 LocID

LocID is a 4-digit alphanumeric code furnished to registered merchants. For sample codes and for using the test server, the value '9502' or '9505' is commonly used.

3.3.3 ACHFileName

ACHFileName is the unique filename of the batch with the following format:

```
        SSSLLLL-YYmmddAA +  '.XML',
            where SSS is the SSS field (see 3.3.1 above)
                   LLL is the LocID field (3.3.2 above)
                   YYmmdd (string format of the date of sending the file)
                   AA is an alphanumeric modifier to the name we recommend iterating it
                      (AA, AB, AC, A9 ... BA, BB, BC, B9 ... ZZ, Z9) etc.
```

Important Note: If you want the system to automatically create a unique filename for you, leave the ACHFileName blank.

3.3.4 TotalNumRecords

TotalNumRecords is the total number of records (ACHTransRecord) in the batch file.

3.3.5 TotalDebitRecords

TotalDebitRecords is the total number of debit transaction records (ACHTransRecord) in the batch file.

3.3.6 TotalCreditRecords
TotalCreditRecords is the total number of credit transaction records (ACHTransRecord) in the batch file.

3.3.7 TotalDebitAmount
TotalDebitAmount is the total amount of debit transaction records (ACHTransRecord) in the batch file.

3.3.7 TotalCreditAmount
TotalCreditAmount is the total amount of credit transaction records (ACHTransRecord) in the batch file.

3.3.8 ACHRecords

ACHRecords is the list or array of ACHTransRecord's in the batch file (ArrayOfACHTransRecord). See Section 3.2 above for the data structure of ACHTransRecord. Again, the use of this field is best understood by looking at the WSDL and/or the classes generated from WSDL; and looking at the sample codes given in Appendix A2 of this guide.

3.4 ACHReturnRecord

The ACHReturnRecord represents the basic or single ACH return and/or settlement record. It is a unit record for an ACH return or settlement e.g. using the GetACHReturns method. It consists of SSS, LocID, SourceFile, FrontEndTrace, ResponseCode, CustTransType, BackEndSN, CustomerName, TransAmount, EffectiveDate, ActionDate and ActionDetail which will be described below. For details on the field types and format please refer to Table 3.2.

3.4.1 SSS
SSS is a 3-letter code furnished to registered merchants. For sample codes and for using the test server, the value 'TST' is commonly used.

3.4.2 LocID
LocID is a 4-digit alphanumeric code furnished to registered merchants. For sample codes and for using the test server, the value '9502' or '9505' is commonly used.

3.4.3 FrontEndTrace
The FrontEndTrace is tracking ID of this particular record or transaction when it was sent or submitted to the ACH.

3.4.4 SourceFile
This is the source file name of the returns. This is for internal use and can be used for tracking purposes.

3.4.5 ResponseCode
The response codes values are defined and described as follows:

Value	Description
1SNT:	The transaction was sent to the ACH system
2STL:	This transaction was settled; i.e. as a result of this transaction: either (a) money was credited to you or your customer, depending on transaction type; or(b) money was credited or debited to you as a result of a return on this transaction;
3RET:	A return from the ACH was received for this transaction. The ActionDetail (Section 3.5.11 below) contains the Return Code.
4INT:	T$$/ACHWorks determined that this transaction cannot be processed and is returned to you. The The ActionDetail (Section 3.5.11 below) contains the Internal Return Code.
5COR:	A correction or notice of change was received from the ACH. The ActionDetail (Section 3.5.11 below) contains the notice of change or correction code.
9BNK:	This represent Total Debit or Credit on your account as a result of various settlements (2STL. The 9BNK transaction should match your bank statement exactly and can be used to reconcile your bank statement with your ACH transactions.

3.4.6 CustTransType
CustTransType is the transaction type. The value is either 'D' for debit customer account or 'C' for credit customer account. If the ResponseCode value is 9BNK, CustTransType is set to blank.

3.4.7 CustomerName
CustomerName is the name of the customer. The value, when transaction is first sent, is trimmed to the maximum length allowed by the ACH which 22 characters.

3.4.8 TransAmount
TransAmount is the amount of the transaction in US dollars, e.g. 575.45

3.4.9 EffectiveDate
EffectiveDate is the date when the transaction was first submitted to the ACH system.

3.4.10 ActionDate
ActionDate is the date which has a value depending on the ResponseCode (Section 3.4.5). It is very useful to the user in determining the order of events when reviewing or investigating a transaction.

ResponseCode	ActionDate
1SNT:	Date transaction is sent to ACH
2STL:	Date transaction was settled
3RET:	Date transaction was returned bay ACH Receiver Bank
4INT:	Date transaction was not processed and returned
5COR:	Date correction was received from the ACH

9BNK:	Date money was credited or debited to your bank

3.4.11 ActionDetail

ActionDetail contains a string of 32 characters and has a value depending on the ResponseCode (Section 3.4.5):

ResponseCode **ActionDetail**

1SNT: This field is blank

2STL: Contains a settlement ID by which this transaction was settled, and a debit or credit was done into your bank account. The format for this is:

"AAAAAAAA BBBBBB CC DD EEEEE" , where

- **AAAAAAAA** = 8 chars, is the settlement ID which also appears on your Bank statement. You can use this to reconcile with your bank Statement.
- **BBBBBB** = 6 chars, either "DEBIT" or "CREDIT" which refers to DEBIT/ CREDIT to your bank account
- **CC** = 2 chars, your AccountSet (see Section 3.2.18) by which this transaction was settled. If this is blank, look for the value in EEEEE
- **DD** = 2 chars, either "TS" or "RS". TS refers to settlement of this Transaction when it clears. RS is when a late return was received on a transaction which was previously settled.
- **EEEEE** = your AccountSet (see Section 3.2.18) by which this transaction was settled.

3RET: Contains the Return Code from ACH. Some of the most common Return Codes are:

R01 – Insufficient Funds
R02 – Closed Account
R03 – No Account
R04 - Invalid Account Number
R09 – Uncollected Funds

Please contact T$$/ACHWorks for a complete list of Return Codes.

4INT: Contains the Internal Return Code from T$$/ACHWorks. Some of the most common Internal Return Codes are:

X01 – Insufficient Funds
X02 – Closed Account
X03 – No Account
X04 - Invalid Account Number

Please contact T$$/ACHWorks for a complete list of Internal Return Codes.

5COR: Contains the correction code and values from the ACH. Of the 32 characters of the ActionDetail, the first 3 characters is the Correction Code. The remaining 29 characters contains the correction value. If the Correction Code is:

CO3 (Incorrect Routing and Acct No), the correction routing number is contained from the 4th to the 12th character. The correction account number is contained from the 16th to 32nd (the last character).
CO6, the correction account number is contained from the 4th to the 20th character. The correction transaction code is contained from 24th to 25th character.
CO7, the correction routing number is contained from the 4th to the 12th character. The correction account number is contained from the 13th to the 29th character. The correction transaction code is contained from 30th to 31st character.

Please contact T$$/ACHWorks for a complete list of Correction Codes.

9BNK: Refer to a settlement ID that would also appear on your bank statement. This corresponds to a collection of your transactions (2STL) with the same settlement IDs. Use this to reconcile with your bank statement.

3.5 ACHReturns

The ACHReturns represents the file with elements or fields that describe an ACH return file. It contains one or more records of ACH returns as ACHReturnRecords (same as the list or array of ACHReturnRecord's). It serves as the output in getting ACH returns i.e. using the GetACHReturns or GetACHReturnsHist method. It consists of SSS, LocID, Status, Details, TotalNumRecords, ReturnDateFrom, ReturnDateTo, TotalNumErrors, Errors, and ACHReturnRecords which will be described below.

3.5.1 SSS

SSS is a 3-letter code furnished to registered merchants.

3.5.2 LocID

LocID is a 4-digit alphanumeric code furnished to registered merchants.

3.5.3 Status

The Status describes the result of calling the GetACHReturns and GetACHReturnsHist methods.

3.5.4 Details

This field desribe the details of the Status when calling the GetACHReturns and GetACHReturnsHist methods.

3.5.5 TotalNumRecords
This is the total number of returns or ACHReturnRecords.

3.5.6 ReturnDateFrom
The specified start return date for getting a set of ACHReturnRecords for this LocID.

3.5.7 ReturnDateTo
The specified end return date for getting a set of ACHReturnRecords for this LocID.

3.5.8 TotalNumErrors
This is the total number of errors encountered as a result of getting the ACHReturnRecords.

3.5.9 Errors
List of errors as an array of String.

3.5.10 ACHReturnRecords
List of all the ACH returns and/or settlements as an array of ACHReturnRecord (ArrayOfACHReturnRecord). See Section 3.4 above for the data structure of ACHReturnRecord. Again, this field is best understood by looking at the WSDL and/or the classes generated from WSDL; and looking at the sample codes given in Appendix A2 of this guide.

3.6 TransResult

TransResult is the data structure of the result or response when calling send methods or operations e.g. SendACHTrans or SendACHTransBatch. It consists of SSS, LocID, Status, Details, CallMethod, FileName, FrontEndTrace, TotalNumErrors and Errors which will be described below.

3.6.1 SSS
SSS is a 3-letter code furnished to registered merchants.   This value is returned when doing a 'send' method call.

3.6.2 LocID
LocID is a 4-digit alphanumeric code furnished to registered merchants. This value is returned when doing a 'send' method call.

3.6.3 Status
This field shows the connection status of doing a 'send' method call. The value is either a 'SUCCESS' or a 'REJECT'.

3.6.4 Details
This field describe in detail the conection status in doing a 'send' method call.
Example:

'Rejected due to errors 'or 'Transaction records received and queued on 04/12/2010 12:56:34 PM Pacific Time [ACHFileName=TST9505-10041200.XML]. Server Processing Time:15.6245 ms'

3.6.5 CallMethod
This returns the 'send' method used i.e. SendACHTrans, SendACHTransBatch or SendACHTransBatchBIN.

3.6.6 CallDateTime
This returns the time stamp of the 'send' method call.

3.6.7 FileName
This field returns the FileName used or generated for doing a 'send' method call.
e.g. TST9505-10041200.XML

3.6.8 FrontEndTrace
This field returns the FrontEndTrace used for doing a SendACHTrans method call.

3.6.9 TotalNumErrors
This field returns the number of errors encountered in a 'REJECTED' send method call. This value is zero if the Status is a 'SUCCESS'.

3.6.10 Errors
List of all the Errors encountered in a 'REJECTED' send method call.
Example:

Error#1:FrontEndTrace X9000123310456 already used.
Error#2:Unable to complete processing of data due to error(s).

3.7 ResultFile

The ResultFile represents the file with elements or fields that describe a list of TransResult from specified data range. It contains one or more records of TransResults (same as the list of TransResult or ArrayOfTransResult). It serves as the output in getting past connection Results and Errors i.e. using the GetResultFile method. It consists of SSS, LocID, Status, Details, TotalResultRecords, ResultDateFrom, ResultDateTo, and TransResults which will be described below.

3.7.1 SSS
SSS is a 3-letter code furnished to registered merchants.

3.7.2 LocID
SSS is a 3-letter code furnished to registered merchants.

3.7.3 Status
This is the Status of calling the GetResultFile method, either a 'REJECTED' or 'SUCCESS' call.

3.7.4 Details
This field describe in detail the Status of calling the GetResultFile.
Example:

```
Retrieved 79 connection records on 04/12/2010 01:16:42 PM Pacific Time. Server Processing Time:<0.0001 ms
```

3.7.5 TotalResultRecords
The total number or result records or array count of TransResults .

3.7.6 ResultDateFrom
The specified start result date for getting a set of TransResults for this LocID.

3.7.7 ResultDateTo
The specified end result date for getting a set of TransResults for this LocID.

3.7.8 TransResults
List of all the transaction results as an array of TransResult (ArrayOfTransResult). See Section 3.6 above for the data structure of TransResult. Again, this field is best understood by looking at the WSDL and/or the classes generated from WSDL; and looking at the sample codes given in Appendix A2 of this guide.

3.8 ErrorRecord

ErrorRecord is the data structure of an error when calling methods or operations e.g. SendACHTrans or SendACHTransBatch. It consists of SSS, LocID, Status, Details, CallMethod, CallDateTime, FileName, FrontEndTrace, TotalNumErrors and Errors which will be described below.

3.8.1 SSS
SSS is a 3-letter code furnished to registered merchants.   This field shows the value of SSS when the error occurred.

3.8.2 LocID
SSS is a 3-letter code furnished to registered merchants.   This field shows the value of LocID when the error occurred.

3.8.3 CallMethod
This field shows the method used when the error occured.

3.8.4 CallDateTime
This returns the time stamp of the error when it occured.

3.8.5 FileName
The FileName used or generated in the method (if applicable) when the error occured.

3.8.6 FrontEndTrace
The FrontEndTrace used in the method (if applicable) when the error occured.

3.8.7 TotalNumErrors
This is the total number of errors encountered as a result of using the 'CallMethod'.

3.8.8 Errors
List of all the errors encountered as a result of a 'REJECTED' CallMethod.

3.9 ErrorFile

The ErrorFile represents the file with elements or fields that describe a list of ErrorRecord from specified data range. It contains one or more records of ErrorRecords (same as the list of ErrorRecord or ArrayOfErrorRecord). It serves as the output in getting past connection Errors i.e. using the GetErrorFile method. It consists of SSS, LocID, Status, Details, TotalErrorRecords, ErrorDateFrom, ErrorDateTo, and ErrorRecords which will be described below.

3.9.1 SSS
SSS is a 3-letter code furnished to registered merchants

3.9.2 LocID
SSS is a 3-letter code furnished to registered merchants.

3.9.3 Status
This is the Status of calling the GetErrorFile method, either a 'REJECTED' or 'SUCCESS' call.

3.9.4 Details
This field describe in detail the Status of calling the GetErrorFile.
Example:

```
Retrieved 9 error records on 04/12/2010 01:16:42 PM Pacific Time. Server Processing Time:<0.0001 ms
```

3.9.5 TotalErrorRecords
The total number or error records or array count of ErrorRecords .

3.9.6 ErrorDateFrom
The specified start error date for getting a set of ErrorRecords for this LocID.

3.9.7 ErrorDateTo
The specified end error date for getting a set of ErrorRecords for this LocID.

3.9.8 ErrorRecords
List of all the errors as an array of ErrorRecord (ArrayOfErrorRecord). See Section 3.8 above for the data structure of ErrorRecord. Again, this field is best understood by looking at the WSDL and/or the classes generated from WSDL; and looking at the sample codes given in Appendix A2 of this guide.

3.10 Field Types and Length

Listed in the table below are the fields with the required types and lengths for data entry/output. The web service allows entry of some of the string types e.g. OriginatorName or CustomerName to exceed the required length but the value is trimmed to the required length during processing and final storage.

Table 3.1 Fields with required types and lengths.

Field	Type	Length	Remarks
SSS	String	3	Use the value assigned by T$$
LocID	String	4	Use the value assigned by T$$
FrontEndTrace	String	12	First character should not be 'W'
OriginatorName	String	16	Originator Name (Individual or Company)
TransactionCode	String	3	Value is either PPD, CCD, TEL, WEB, RCK, POS, etc.
CustTransType	String	1	Put 'D' for Debit, 'C' for Credit
CustomerID	String	12	Optional field for ACHWorks-WEB
CustomerName	String	22	Format: Lastname, Firstname or Companyname
CustomerRoutingNo	String	9	Bank Routing No
CustomerAcctNo	String	17	Include prefix and suffix 0's, include hyphens
CustomerAcctType	String	1	Put 'C' for Checking, 'S' for Savings
TransAmount	Double	-	Amount in dollars, e.g. 100.75, MAX = 999999.99
CheckOrCustID	String	15	Check number or Customer ID
CheckOrTransDate	ns:dateTime	-	Value should be <= EffectiveDate
EffectiveDate	ns:dateTime	-	Value should be >= CheckOrTransDate
Memo	String	10	This appears on customer's bank statement
OpCode	String	1	Put 'R' if recurring, Default value is 'S'
AccountSet	String	1	Use the value(s) assigned by T$$
ACHFileName	String	20	Format: SSSLLLL-YYmmddAA.XML

4. Contact Information

For more information and/or questions please email ACHWorks support: support@achworks.com or call us at +1 916 638 8811.

APPENDICES

A1. Service Description

Test WSDL URL: http://tstsvr.achworks.com/dnet/achws.asmx?WSDL
(A Live WSDL URL will be provided to registered merchants)

```
  <?xml version="1.0" encoding="utf-8" ?>
- <wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:tns="http://achworks.com/"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/" targetNamespace="http://achworks.com/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">ACHWorksWS is a web service
(web API) by <a href="http://www.achworks.com">ACHWorks</a> that provides a complete solution for
Automated Clearing House (ACH) operations: Sending transactions and getting status of returns and
settlements. Please click <a href="http://www.achworks.com/contactus">here</a> to contact us for more
information. Last updated on: 03/25/2010. Beta Version.</wsdl:documentation>
- <wsdl:types>
- <s:schema elementFormDefault="qualified" targetNamespace="http://achworks.com/">
- <s:element name="ConnectionCheck">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="InpCompanyInfo" type="tns:CompanyInfo" />
  </s:sequence>
  </s:complexType>
  </s:element>
- <s:complexType name="CompanyInfo">
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="SSS" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="LocID" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="Company" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CompanyKey" type="s:string" />
  </s:sequence>
  </s:complexType>
- <s:element name="ConnectionCheckResponse">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="ConnectionCheckResult" type="s:string" />
  </s:sequence>
  </s:complexType>
  </s:element>
- <s:element name="SendACHTrans">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="InpCompanyInfo" type="tns:CompanyInfo" />
  <s:element minOccurs="0" maxOccurs="1" name="InpACHTransRecord" type="tns:ACHTransRecord" />
  </s:sequence>
  </s:complexType>
  </s:element>
- <s:complexType name="ACHTransRecord">
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="SSS" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="LocID" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="FrontEndTrace" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="OriginatorName" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="TransactionCode" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustTransType" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustomerID" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustomerName" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustomerRoutingNo" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustomerAcctNo" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustomerAcctType" type="s:string" />
  <s:element minOccurs="1" maxOccurs="1" name="TransAmount" type="s:double" />
  <s:element minOccurs="0" maxOccurs="1" name="CheckOrCustID" type="s:string" />
  <s:element minOccurs="1" maxOccurs="1" name="CheckOrTransDate" type="s:dateTime" />
  <s:element minOccurs="1" maxOccurs="1" name="EffectiveDate" type="s:dateTime" />
  <s:element minOccurs="0" maxOccurs="1" name="Memo" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="OpCode" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="AccountSet" type="s:string" />
  </s:sequence>
  </s:complexType>
- <s:element name="SendACHTransResponse">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="SendACHTransResult" type="tns:TransResult" />
  </s:sequence>
  </s:complexType>
  </s:element>
- <s:complexType name="TransResult">
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="SSS" type="s:string" />
```

```
 <s:element minOccurs="0" maxOccurs="1" name="LocID" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="Status" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="Details" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="CallMethod" type="s:string" />
 <s:element minOccurs="1" maxOccurs="1" name="CallDateTime" type="s:dateTime" />
 <s:element minOccurs="0" maxOccurs="1" name="FileName" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="FrontEndTrace" type="s:string" />
 <s:element minOccurs="1" maxOccurs="1" name="TotalNumErrors" type="s:int" />
 <s:element minOccurs="0" maxOccurs="1" name="Errors" type="tns:ArrayOfString" />
 </s:sequence>
 </s:complexType>
- <s:complexType name="ArrayOfString">
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="unbounded" name="string" nillable="true" type="s:string" />
 </s:sequence>
 </s:complexType>
- <s:element name="SendACHTransBatch">
- <s:complexType>
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="1" name="InpCompanyInfo" type="tns:CompanyInfo" />
 <s:element minOccurs="0" maxOccurs="1" name="InpACHFile" type="tns:ACHFile" />
 </s:sequence>
 </s:complexType>
 </s:element>
- <s:complexType name="ACHFile">
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="1" name="SSS" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="LocID" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="ACHFileName" type="s:string" />
 <s:element minOccurs="1" maxOccurs="1" name="TotalNumRecords" type="s:int" />
 <s:element minOccurs="1" maxOccurs="1" name="TotalDebitRecords" type="s:int" />
 <s:element minOccurs="1" maxOccurs="1" name="TotalDebitAmount" type="s:double" />
 <s:element minOccurs="1" maxOccurs="1" name="TotalCreditRecords" type="s:int" />
 <s:element minOccurs="1" maxOccurs="1" name="TotalCreditAmount" type="s:double" />
 <s:element minOccurs="0" maxOccurs="1" name="ACHRecords" type="tns:ArrayOfACHTransRecord" />
 </s:sequence>
 </s:complexType>
- <s:complexType name="ArrayOfACHTransRecord">
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="unbounded" name="ACHTransRecord" nillable="true"
type="tns:ACHTransRecord" />
 </s:sequence>
 </s:complexType>
- <s:element name="SendACHTransBatchResponse">
- <s:complexType>
- <s:sequence>
```

```
  <s:element minOccurs="0" maxOccurs="1" name="SendACHTransBatchResult" type="tns:TransResult" />
  </s:sequence>
  </s:complexType>
  </s:element>
- <s:element name="SendACHTransBatchBin">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="InpCompanyInfo" type="tns:CompanyInfo" />
  <s:element minOccurs="0" maxOccurs="1" name="InpBinFile" type="s:base64Binary" />
  </s:sequence>
  </s:complexType>
  </s:element>
- <s:element name="SendACHTransBatchBinResponse">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="SendACHTransBatchBinResult" type="tns:TransResult" />
  </s:sequence>
  </s:complexType>
  </s:element>
- <s:element name="SendIATACHTransBatch">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="InpCompanyInfo" type="tns:CompanyInfo" />
  <s:element minOccurs="0" maxOccurs="1" name="InpACHFile" type="tns:IATACHFile" />
  </s:sequence>
  </s:complexType>
  </s:element>
- <s:complexType name="IATACHFile">
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="SSS" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="LocID" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="ACHFileName" type="s:string" />
  <s:element minOccurs="1" maxOccurs="1" name="TotalNumRecords" type="s:int" />
  <s:element minOccurs="1" maxOccurs="1" name="TotalDebitRecords" type="s:int" />
  <s:element minOccurs="1" maxOccurs="1" name="TotalDebitAmount" type="s:double" />
  <s:element minOccurs="1" maxOccurs="1" name="TotalCreditRecords" type="s:int" />
  <s:element minOccurs="1" maxOccurs="1" name="TotalCreditAmount" type="s:double" />
  <s:element minOccurs="0" maxOccurs="1" name="IATOptions" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="IATACHRecords" type="tns:ArrayOfIATACHTransRecord"
/>
  </s:sequence>
  </s:complexType>
- <s:complexType name="ArrayOfIATACHTransRecord">
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="unbounded" name="IATACHTransRecord" nillable="true"
type="tns:IATACHTransRecord" />
```

```
  </s:sequence>
  </s:complexType>
- <s:complexType name="IATACHTransRecord">
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="SSS" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="LocID" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="FrontEndTrace" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="OriginatorName" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="TransactionCode" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustTransType" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustomerID" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustomerName" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustomerRoutingNo" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustomerAcctNo" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustomerAcctType" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustomerAddress" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustomerBank" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustomerBankAddress" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustomerBenef" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CustomerBenefAddress" type="s:string" />
  <s:element minOccurs="1" maxOccurs="1" name="TransAmount" type="s:double" />
  <s:element minOccurs="0" maxOccurs="1" name="CheckOrCustID" type="s:string" />
  <s:element minOccurs="1" maxOccurs="1" name="CheckOrTransDate" type="s:dateTime" />
  <s:element minOccurs="1" maxOccurs="1" name="EffectiveDate" type="s:dateTime" />
  <s:element minOccurs="0" maxOccurs="1" name="Memo" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="OpCode" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="AccountSet" type="s:string" />
  </s:sequence>
  </s:complexType>
- <s:element name="SendIATACHTransBatchResponse">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="SendIATACHTransBatchResult" type="tns:TransResult" />
  </s:sequence>
  </s:complexType>
  </s:element>
- <s:element name="GetACHReturns">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="InpCompanyInfo" type="tns:CompanyInfo" />
  </s:sequence>
  </s:complexType>
  </s:element>
- <s:element name="GetACHReturnsResponse">
- <s:complexType>
- <s:sequence>
```

```
 <s:element minOccurs="0" maxOccurs="1" name="GetACHReturnsResult" type="tns:ACHReturns" />
 </s:sequence>
 </s:complexType>
 </s:element>
- <s:complexType name="ACHReturns">
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="1" name="SSS" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="LocID" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="Status" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="Details" type="s:string" />
 <s:element minOccurs="1" maxOccurs="1" name="TotalNumRecords" type="s:int" />
 <s:element minOccurs="1" maxOccurs="1" name="ReturnDateFrom" nillable="true" type="s:dateTime" />
 <s:element minOccurs="1" maxOccurs="1" name="ReturnDateTo" nillable="true" type="s:dateTime" />
 <s:element minOccurs="1" maxOccurs="1" name="TotalNumErrors" type="s:int" />
 <s:element minOccurs="0" maxOccurs="1" name="Errors" type="tns:ArrayOfString" />
 <s:element minOccurs="0" maxOccurs="1" name="ACHReturnRecords"
type="tns:ArrayOfACHReturnRecord" />
 </s:sequence>
 </s:complexType>
- <s:complexType name="ArrayOfACHReturnRecord">
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="unbounded" name="ACHReturnRecord" nillable="true"
type="tns:ACHReturnRecord" />
 </s:sequence>
 </s:complexType>
- <s:complexType name="ACHReturnRecord">
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="1" name="SSS" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="LocID" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="SourceFile" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="FrontEndTrace" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="ResponseCode" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="CustTransType" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="BackEndSN" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="CustomerName" type="s:string" />
 <s:element minOccurs="1" maxOccurs="1" name="TransAmount" nillable="true" type="s:double" />
 <s:element minOccurs="1" maxOccurs="1" name="EffectiveDate" nillable="true" type="s:dateTime" />
 <s:element minOccurs="1" maxOccurs="1" name="ActionDate" nillable="true" type="s:dateTime" />
 <s:element minOccurs="0" maxOccurs="1" name="ActionDetail" type="s:string" />
 </s:sequence>
 </s:complexType>
- <s:element name="GetACHReturnsHist">
- <s:complexType>
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="1" name="InpCompanyInfo" type="tns:CompanyInfo" />
 <s:element minOccurs="1" maxOccurs="1" name="ReturnDateFrom" type="s:dateTime" />
 <s:element minOccurs="1" maxOccurs="1" name="ReturnDateTo" type="s:dateTime" />
 </s:sequence>
 </s:complexType>
 </s:element>
- <s:element name="GetACHReturnsHistResponse">
- <s:complexType>
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="1" name="GetACHReturnsHistResult" type="tns:ACHReturns" />
 </s:sequence>
 </s:complexType>
 </s:element>
- <s:element name="GetACHReturnsBin">
- <s:complexType>
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="1" name="InpCompanyInfo" type="tns:CompanyInfo" />
 <s:element minOccurs="1" maxOccurs="1" name="InpReturnFormat" type="tns:ReturnsFormat" />
 </s:sequence>
 </s:complexType>
 </s:element>
- <s:simpleType name="ReturnsFormat">
- <s:restriction base="s:string">
 <s:enumeration value="ByRecord" />
 <s:enumeration value="ByFile" />
 </s:restriction>
 </s:simpleType>
- <s:element name="GetACHReturnsBinResponse">
- <s:complexType>
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="1" name="GetACHReturnsBinResult" type="s:base64Binary" />
 </s:sequence>
 </s:complexType>
 </s:element>
- <s:element name="GetACHReturnsHistBin">
- <s:complexType>
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="1" name="InpCompanyInfo" type="tns:CompanyInfo" />
 <s:element minOccurs="1" maxOccurs="1" name="ReturnDateFrom" type="s:dateTime" />
 <s:element minOccurs="1" maxOccurs="1" name="ReturnDateTo" type="s:dateTime" />
 <s:element minOccurs="1" maxOccurs="1" name="InpReturnFormat" type="tns:ReturnsFormat" />
 </s:sequence>
 </s:complexType>
 </s:element>
- <s:element name="GetACHReturnsHistBinResponse">
- <s:complexType>
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="1" name="GetACHReturnsHistBinResult" type="s:base64Binary" />
 </s:sequence>
 </s:complexType>
 </s:element>
- <s:element name="GetResultFile">
- <s:complexType>
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="1" name="InpCompanyInfo" type="tns:CompanyInfo" />
 <s:element minOccurs="1" maxOccurs="1" name="ResultDateFrom" type="s:dateTime" />
 <s:element minOccurs="1" maxOccurs="1" name="ResultDateTo" type="s:dateTime" />
 </s:sequence>
 </s:complexType>
 </s:element>
- <s:element name="GetResultFileResponse">
- <s:complexType>
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="1" name="GetResultFileResult" type="tns:ResultFile" />
 </s:sequence>
 </s:complexType>
 </s:element>
- <s:complexType name="ResultFile">
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="1" name="SSS" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="LocID" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="Status" type="s:string" />
 <s:element minOccurs="0" maxOccurs="1" name="Details" type="s:string" />
 <s:element minOccurs="1" maxOccurs="1" name="TotalResultRecords" type="s:int" />
 <s:element minOccurs="1" maxOccurs="1" name="ResultDateFrom" nillable="true" type="s:dateTime" />
 <s:element minOccurs="1" maxOccurs="1" name="ResultDateTo" nillable="true" type="s:dateTime" />
 <s:element minOccurs="0" maxOccurs="1" name="TransResults" type="tns:ArrayOfTransResult" />
 </s:sequence>
 </s:complexType>
- <s:complexType name="ArrayOfTransResult">
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="unbounded" name="TransResult" nillable="true"
type="tns:TransResult" />
 </s:sequence>
 </s:complexType>
- <s:element name="GetErrorFile">
- <s:complexType>
- <s:sequence>
 <s:element minOccurs="0" maxOccurs="1" name="InpCompanyInfo" type="tns:CompanyInfo" />
 <s:element minOccurs="1" maxOccurs="1" name="ErrorDateFrom" type="s:dateTime" />
 <s:element minOccurs="1" maxOccurs="1" name="ErrorDateTo" type="s:dateTime" />
 </s:sequence>
 </s:complexType>
 </s:element>
```

```
- <s:element name="GetErrorFileResponse">
- <s:complexType>
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="GetErrorFileResult" type="tns:ErrorFile" />
  </s:sequence>
  </s:complexType>
  </s:element>
- <s:complexType name="ErrorFile">
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="SSS" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="LocID" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="Status" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="Details" type="s:string" />
  <s:element minOccurs="1" maxOccurs="1" name="TotalErrorRecords" type="s:int" />
  <s:element minOccurs="1" maxOccurs="1" name="ErrorDateFrom" nillable="true" type="s:dateTime" />
  <s:element minOccurs="1" maxOccurs="1" name="ErrorDateTo" nillable="true" type="s:dateTime" />
  <s:element minOccurs="0" maxOccurs="1" name="ErrorRecords" type="tns:ArrayOfErrorRecord" />
  </s:sequence>
  </s:complexType>
- <s:complexType name="ArrayOfErrorRecord">
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="unbounded" name="ErrorRecord" nillable="true"
type="tns:ErrorRecord" />
  </s:sequence>
  </s:complexType>
- <s:complexType name="ErrorRecord">
- <s:sequence>
  <s:element minOccurs="0" maxOccurs="1" name="SSS" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="LocID" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="CallMethod" type="s:string" />
  <s:element minOccurs="1" maxOccurs="1" name="CallDateTime" type="s:dateTime" />
  <s:element minOccurs="0" maxOccurs="1" name="FileName" type="s:string" />
  <s:element minOccurs="0" maxOccurs="1" name="FrontEndTrace" type="s:string" />
  <s:element minOccurs="1" maxOccurs="1" name="TotalNumErrors" type="s:int" />
  <s:element minOccurs="0" maxOccurs="1" name="Errors" type="tns:ArrayOfString" />
  </s:sequence>
  </s:complexType>
  </s:schema>
  </wsdl:types>
- <wsdl:message name="ConnectionCheckSoapIn">
  <wsdl:part name="parameters" element="tns:ConnectionCheck" />
  </wsdl:message>
- <wsdl:message name="ConnectionCheckSoapOut">
  <wsdl:part name="parameters" element="tns:ConnectionCheckResponse" />
  </wsdl:message>
- <wsdl:message name="SendACHTransSoapIn">
```

```
  <wsdl:part name="parameters" element="tns:SendACHTrans" />
  </wsdl:message>
- <wsdl:message name="SendACHTransSoapOut">
  <wsdl:part name="parameters" element="tns:SendACHTransResponse" />
  </wsdl:message>
- <wsdl:message name="SendACHTransBatchSoapIn">
  <wsdl:part name="parameters" element="tns:SendACHTransBatch" />
  </wsdl:message>
- <wsdl:message name="SendACHTransBatchSoapOut">
  <wsdl:part name="parameters" element="tns:SendACHTransBatchResponse" />
  </wsdl:message>
- <wsdl:message name="SendACHTransBatchBinSoapIn">
  <wsdl:part name="parameters" element="tns:SendACHTransBatchBin" />
  </wsdl:message>
- <wsdl:message name="SendACHTransBatchBinSoapOut">
  <wsdl:part name="parameters" element="tns:SendACHTransBatchBinResponse" />
  </wsdl:message>
- <wsdl:message name="SendIATACHTransBatchSoapIn">
  <wsdl:part name="parameters" element="tns:SendIATACHTransBatch" />
  </wsdl:message>
- <wsdl:message name="SendIATACHTransBatchSoapOut">
  <wsdl:part name="parameters" element="tns:SendIATACHTransBatchResponse" />
  </wsdl:message>
- <wsdl:message name="GetACHReturnsSoapIn">
  <wsdl:part name="parameters" element="tns:GetACHReturns" />
  </wsdl:message>
- <wsdl:message name="GetACHReturnsSoapOut">
  <wsdl:part name="parameters" element="tns:GetACHReturnsResponse" />
  </wsdl:message>
- <wsdl:message name="GetACHReturnsHistSoapIn">
  <wsdl:part name="parameters" element="tns:GetACHReturnsHist" />
  </wsdl:message>
- <wsdl:message name="GetACHReturnsHistSoapOut">
  <wsdl:part name="parameters" element="tns:GetACHReturnsHistResponse" />
  </wsdl:message>
- <wsdl:message name="GetACHReturnsBinSoapIn">
  <wsdl:part name="parameters" element="tns:GetACHReturnsBin" />
  </wsdl:message>
- <wsdl:message name="GetACHReturnsBinSoapOut">
  <wsdl:part name="parameters" element="tns:GetACHReturnsBinResponse" />
  </wsdl:message>
- <wsdl:message name="GetACHReturnsHistBinSoapIn">
  <wsdl:part name="parameters" element="tns:GetACHReturnsHistBin" />
  </wsdl:message>
- <wsdl:message name="GetACHReturnsHistBinSoapOut">
  <wsdl:part name="parameters" element="tns:GetACHReturnsHistBinResponse" />
  </wsdl:message>
- <wsdl:message name="GetResultFileSoapIn">
  <wsdl:part name="parameters" element="tns:GetResultFile" />
  </wsdl:message>
- <wsdl:message name="GetResultFileSoapOut">
  <wsdl:part name="parameters" element="tns:GetResultFileResponse" />
  </wsdl:message>
- <wsdl:message name="GetErrorFileSoapIn">
  <wsdl:part name="parameters" element="tns:GetErrorFile" />
  </wsdl:message>
- <wsdl:message name="GetErrorFileSoapOut">
  <wsdl:part name="parameters" element="tns:GetErrorFileResponse" />
  </wsdl:message>
- <wsdl:portType name="ACHWorksWSSoap">
- <wsdl:operation name="ConnectionCheck">
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Checks validity of user/company
information</wsdl:documentation>
  <wsdl:input message="tns:ConnectionCheckSoapIn" />
  <wsdl:output message="tns:ConnectionCheckSoapOut" />
  </wsdl:operation>
- <wsdl:operation name="SendACHTrans">
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Sends single ACH transaction
record</wsdl:documentation>
  <wsdl:input message="tns:SendACHTransSoapIn" />
  <wsdl:output message="tns:SendACHTransSoapOut" />
  </wsdl:operation>
- <wsdl:operation name="SendACHTransBatch">
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Sends multiple ACH transaction
records</wsdl:documentation>
  <wsdl:input message="tns:SendACHTransBatchSoapIn" />
  <wsdl:output message="tns:SendACHTransBatchSoapOut" />
  </wsdl:operation>
- <wsdl:operation name="SendACHTransBatchBin">
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Sends multiple ACH transaction
records in a binary file</wsdl:documentation>
  <wsdl:input message="tns:SendACHTransBatchBinSoapIn" />
  <wsdl:output message="tns:SendACHTransBatchBinSoapOut" />
  </wsdl:operation>
- <wsdl:operation name="SendIATACHTransBatch">
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Sends multiple IAT ACH transaction
records</wsdl:documentation>
  <wsdl:input message="tns:SendIATACHTransBatchSoapIn" />
  <wsdl:output message="tns:SendIATACHTransBatchSoapOut" />
  </wsdl:operation>
- <wsdl:operation name="GetACHReturns">
```

```
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Gets returns and deletes from
server</wsdl:documentation>
  <wsdl:input message="tns:GetACHReturnsSoapIn" />
  <wsdl:output message="tns:GetACHReturnsSoapOut" />
  </wsdl:operation>
- <wsdl:operation name="GetACHReturnsHist">
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Gets returns from given date
range</wsdl:documentation>
  <wsdl:input message="tns:GetACHReturnsHistSoapIn" />
  <wsdl:output message="tns:GetACHReturnsHistSoapOut" />
  </wsdl:operation>
- <wsdl:operation name="GetACHReturnsBin">
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Gets binary return files and deletes
from server</wsdl:documentation>
  <wsdl:input message="tns:GetACHReturnsBinSoapIn" />
  <wsdl:output message="tns:GetACHReturnsBinSoapOut" />
  </wsdl:operation>
- <wsdl:operation name="GetACHReturnsHistBin">
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Gets binary return files from given
date range</wsdl:documentation>
  <wsdl:input message="tns:GetACHReturnsHistBinSoapIn" />
  <wsdl:output message="tns:GetACHReturnsHistBinSoapOut" />
  </wsdl:operation>
- <wsdl:operation name="GetResultFile">
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Gets result records at specified date
range</wsdl:documentation>
  <wsdl:input message="tns:GetResultFileSoapIn" />
  <wsdl:output message="tns:GetResultFileSoapOut" />
  </wsdl:operation>
- <wsdl:operation name="GetErrorFile">
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">Gets error records at specified date
range</wsdl:documentation>
  <wsdl:input message="tns:GetErrorFileSoapIn" />
  <wsdl:output message="tns:GetErrorFileSoapOut" />
  </wsdl:operation>
  </wsdl:portType>
- <wsdl:binding name="ACHWorksWSSoap" type="tns:ACHWorksWSSoap">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
- <wsdl:operation name="ConnectionCheck">
  <soap:operation soapAction="http://achworks.com/ConnectionCheck" style="document" />
- <wsdl:input>
  <soap:body use="literal" />
  </wsdl:input>
- <wsdl:output>
  <soap:body use="literal" />
  </wsdl:output>
```

```
 </wsdl:operation>
- <wsdl:operation name="SendACHTrans">
 <soap:operation soapAction="http://achworks.com/SendACHTrans" style="document" />
- <wsdl:input>
 <soap:body use="literal" />
 </wsdl:input>
- <wsdl:output>
 <soap:body use="literal" />
 </wsdl:output>
 </wsdl:operation>
- <wsdl:operation name="SendACHTransBatch">
 <soap:operation soapAction="http://achworks.com/SendACHTransBatch" style="document" />
- <wsdl:input>
 <soap:body use="literal" />
 </wsdl:input>
- <wsdl:output>
 <soap:body use="literal" />
 </wsdl:output>
 </wsdl:operation>
- <wsdl:operation name="SendACHTransBatchBin">
 <soap:operation soapAction="http://achworks.com/SendACHTransBatchBin" style="document" />
- <wsdl:input>
 <soap:body use="literal" />
 </wsdl:input>
- <wsdl:output>
 <soap:body use="literal" />
 </wsdl:output>
 </wsdl:operation>
- <wsdl:operation name="SendIATACHTransBatch">
 <soap:operation soapAction="http://achworks.com/SendIATACHTransBatch" style="document" />
- <wsdl:input>
 <soap:body use="literal" />
 </wsdl:input>
- <wsdl:output>
 <soap:body use="literal" />
 </wsdl:output>
 </wsdl:operation>
- <wsdl:operation name="GetACHReturns">
 <soap:operation soapAction="http://achworks.com/GetACHReturns" style="document" />
- <wsdl:input>
 <soap:body use="literal" />
 </wsdl:input>
- <wsdl:output>
 <soap:body use="literal" />
 </wsdl:output>
 </wsdl:operation>
```

```
- <wsdl:operation name="GetACHReturnsHist">
  <soap:operation soapAction="http://achworks.com/GetACHReturnsHist" style="document" />
- <wsdl:input>
  <soap:body use="literal" />
  </wsdl:input>
- <wsdl:output>
  <soap:body use="literal" />
  </wsdl:output>
  </wsdl:operation>
- <wsdl:operation name="GetACHReturnsBin">
  <soap:operation soapAction="http://achworks.com/GetACHReturnsBin" style="document" />
- <wsdl:input>
  <soap:body use="literal" />
  </wsdl:input>
- <wsdl:output>
  <soap:body use="literal" />
  </wsdl:output>
  </wsdl:operation>
- <wsdl:operation name="GetACHReturnsHistBin">
  <soap:operation soapAction="http://achworks.com/GetACHReturnsHistBin" style="document" />
- <wsdl:input>
  <soap:body use="literal" />
  </wsdl:input>
- <wsdl:output>
  <soap:body use="literal" />
  </wsdl:output>
  </wsdl:operation>
- <wsdl:operation name="GetResultFile">
  <soap:operation soapAction="http://achworks.com/GetResultFile" style="document" />
- <wsdl:input>
  <soap:body use="literal" />
  </wsdl:input>
- <wsdl:output>
  <soap:body use="literal" />
  </wsdl:output>
  </wsdl:operation>
- <wsdl:operation name="GetErrorFile">
  <soap:operation soapAction="http://achworks.com/GetErrorFile" style="document" />
- <wsdl:input>
  <soap:body use="literal" />
  </wsdl:input>
- <wsdl:output>
  <soap:body use="literal" />
  </wsdl:output>
  </wsdl:operation>
  </wsdl:binding>
```

```
- <wsdl:binding name="ACHWorksWSSoap12" type="tns:ACHWorksWSSoap">
  <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
- <wsdl:operation name="ConnectionCheck">
  <soap12:operation soapAction="http://achworks.com/ConnectionCheck" style="document" />
- <wsdl:input>
  <soap12:body use="literal" />
  </wsdl:input>
- <wsdl:output>
  <soap12:body use="literal" />
  </wsdl:output>
  </wsdl:operation>
- <wsdl:operation name="SendACHTrans">
  <soap12:operation soapAction="http://achworks.com/SendACHTrans" style="document" />
- <wsdl:input>
  <soap12:body use="literal" />
  </wsdl:input>
- <wsdl:output>
  <soap12:body use="literal" />
  </wsdl:output>
  </wsdl:operation>
- <wsdl:operation name="SendACHTransBatch">
  <soap12:operation soapAction="http://achworks.com/SendACHTransBatch" style="document" />
- <wsdl:input>
  <soap12:body use="literal" />
  </wsdl:input>
- <wsdl:output>
  <soap12:body use="literal" />
  </wsdl:output>
  </wsdl:operation>
- <wsdl:operation name="SendACHTransBatchBin">
  <soap12:operation soapAction="http://achworks.com/SendACHTransBatchBin" style="document" />
- <wsdl:input>
  <soap12:body use="literal" />
  </wsdl:input>
- <wsdl:output>
  <soap12:body use="literal" />
  </wsdl:output>
  </wsdl:operation>
- <wsdl:operation name="SendIATACHTransBatch">
  <soap12:operation soapAction="http://achworks.com/SendIATACHTransBatch" style="document" />
- <wsdl:input>
  <soap12:body use="literal" />
  </wsdl:input>
- <wsdl:output>
  <soap12:body use="literal" />
  </wsdl:output>
```

```
 </wsdl:operation>
- <wsdl:operation name="GetACHReturns">
 <soap12:operation soapAction="http://achworks.com/GetACHReturns" style="document" />
- <wsdl:input>
 <soap12:body use="literal" />
 </wsdl:input>
- <wsdl:output>
 <soap12:body use="literal" />
 </wsdl:output>
 </wsdl:operation>
- <wsdl:operation name="GetACHReturnsHist">
 <soap12:operation soapAction="http://achworks.com/GetACHReturnsHist" style="document" />
- <wsdl:input>
 <soap12:body use="literal" />
 </wsdl:input>
- <wsdl:output>
 <soap12:body use="literal" />
 </wsdl:output>
 </wsdl:operation>
- <wsdl:operation name="GetACHReturnsBin">
 <soap12:operation soapAction="http://achworks.com/GetACHReturnsBin" style="document" />
- <wsdl:input>
 <soap12:body use="literal" />
 </wsdl:input>
- <wsdl:output>
 <soap12:body use="literal" />
 </wsdl:output>
 </wsdl:operation>
- <wsdl:operation name="GetACHReturnsHistBin">
 <soap12:operation soapAction="http://achworks.com/GetACHReturnsHistBin" style="document" />
- <wsdl:input>
 <soap12:body use="literal" />
 </wsdl:input>
- <wsdl:output>
 <soap12:body use="literal" />
 </wsdl:output>
 </wsdl:operation>
- <wsdl:operation name="GetResultFile">
 <soap12:operation soapAction="http://achworks.com/GetResultFile" style="document" />
- <wsdl:input>
 <soap12:body use="literal" />
 </wsdl:input>
- <wsdl:output>
 <soap12:body use="literal" />
 </wsdl:output>
 </wsdl:operation>
```

```
- <wsdl:operation name="GetErrorFile">
  <soap12:operation soapAction="http://achworks.com/GetErrorFile" style="document" />
- <wsdl:input>
  <soap12:body use="literal" />
  </wsdl:input>
- <wsdl:output>
  <soap12:body use="literal" />
  </wsdl:output>
  </wsdl:operation>
  </wsdl:binding>
- <wsdl:service name="ACHWorksWS">
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">ACHWorksWS is a web service
(web API) by <a href="http://www.achworks.com">ACHWorks</a> that provides a complete solution for
Automated Clearing House (ACH) operations: Sending transactions and getting status of returns and
settlements. Please click <a href="http://www.achworks.com/contactus">here</a> to contact us for more
information. Last updated on: 03/25/2010. Beta Version.</wsdl:documentation>
- <wsdl:port name="ACHWorksWSSoap" binding="tns:ACHWorksWSSoap">
  <soap:address location="http://tstsvr.achworks.com/dnet/achws.asmx" />
  </wsdl:port>
- <wsdl:port name="ACHWorksWSSoap12" binding="tns:ACHWorksWSSoap12">
  <soap12:address location="http://tstsvr.achworks.com/dnet/achws.asmx" />
  </wsdl:port>
  </wsdl:service>
  </wsdl:definitions>.
```

A2. Sample Client Codes

The sample codes below are written for the most popular and recent development platforms namely PHP, C#, Visual Basic (VB.NET), ASP.NET and JAVA.

Brief descriptions are provided to provide ease in understanding the required input and output and their formats.

Please contact us if you want a digital copy of the sample codes. See Section 4 for contact information.

A2.1 PHP (Version 5)

connectioncheck.php

```
<?php
//ACHWORKS-SOAP Ver4.0 ConnectionCheck (Connection Check)
//company info
class CompanyInfo {
    public $SSS;
    public $LocID;
    public $Company;
    public $CompanyKey;
}

$mycompanyinfo = new CompanyInfo;
$mycompanyinfo -> SSS = "TST";
$mycompanyinfo -> LocID = "9505";
$mycompanyinfo -> Company = "THAT COMPANY";
$mycompanyinfo -> CompanyKey = "RICO";


$myclient = new SoapClient("http://tstsvr.achworks.com/dnet/achws.asmx?WSDL");


//Important:  use InpCompanyInfo in the soap stru not CompanyInfo
$myresult = $myclient->ConnectionCheck(array("InpCompanyInfo"=>$mycompanyinfo));

print($myresult->ConnectionCheckResult);
?>
```

sendachtrans.php

```
<?php
//ACHWORKS-SOAP Ver3.0 SendACHTrans (Sends a single record)
//3.25.2010 - rico pamplona, rpamplonaATachworksDOTcom
//company info
class CompanyInfo {
    public $SSS;
    public $LocID;
    public $Company;
    public $CompanyKey;
}

class ACHTransRecord {
    public $SSS;
    public $LocID;
    public $FrontEndTrace;
    public $OriginatorName;
    public $TransactionCode;
    public $CustTransType;
    public $CustomerID;
    public $CustomerName;
    public $CustomerRoutingNo;
    public $CustomerAcctNo;
    public $CustomerAcctType;
    public $TransAmount;
    public $CheckOrCustID;
    public $CheckOrTransDate;
    public $EffectiveDate;
    public $Memo;
    public $OpCode;
    public $AccountSet;
}


//CompanyInfo
$mycompanyinfo = new CompanyInfo;
$mycompanyinfo -> SSS = "TST";
$mycompanyinfo -> LocID = "9505";
$mycompanyinfo -> Company = "THAT COMPANY";
$mycompanyinfo -> CompanyKey = "RICO";


//ACHTransRecord
$myachtransrecord = new ACHTransRecord;
$myachtransrecord -> SSS = "TST"; //T$$ assigned
$myachtransrecord -> LocID = "9505"; //T$$ assigned
$myachtransrecord -> FrontEndTrace = "W9000123310454"; //impt!!!Needs a unique value each record per LocID
$myachtransrecord -> OriginatorName = "MYCOMPANY";
$myachtransrecord -> TransactionCode = "PPD"; //PPD, CCD, WEB, TEL, RCK, etc.
$myachtransrecord -> CustTransType = "D";  //D for debit, C for credit
$myachtransrecord -> CustomerID = "CustID12235";
$myachtransrecord -> CustomerName = "DOE, JOHNNY";
$myachtransrecord -> CustomerRoutingNo ="987654320";
$myachtransrecord -> CustomerAcctNo = "0000388836291";
```

```
$myachtransrecord -> CustomerAcctType = "C"; //C for checking, S for Savings
$myachtransrecord -> TransAmount = 100.75;
$myachtransrecord -> CheckOrCustID = "9166388811";
$myachtransrecord -> CheckOrTransDate = '2010-03-24';  //include leading zero for mm and dd e.g. 01 for Jan
$myachtransrecord -> EffectiveDate = '2010-03-24';     //include leading zero for mm and dd e.g. 01 for Jan
$myachtransrecord -> Memo = "Payment";
$myachtransrecord -> OpCode = "S"; //S for Single, R for recurring
$myachtransrecord -> AccountSet = "1"; //T$$ assigned


//SOAP call - test server
$myclient = new SoapClient("http://tstsvr.achworks.com/dnet/achws.asmx?WSDL");
$myresult = $myclient->SendACHTrans(array("InpCompanyInfo"=>$mycompanyinfo,"InpACHTransRecord"=>$myachtransrecord))-
                    >SendACHTransResult;

//print status and details
print($myresult->Status . ", " . $myresult->Details . "<br>");

//print errors if there is any
foreach ($myresult->Errors->string as $myerror) {
print($myerror . "<br>");
}
?>
```

sendachtransbatch.php

```
<?php
//ACHWORKS-SOAP Ver3.0 SendACHTransBatch (Sends multiple records)
//3.25.2010 - rico pamplona, rpamplonaATachworksDOTcom
//company info
class CompanyInfo {
    public $SSS;
    public $LocID;
    public $Company;
    public $CompanyKey;
}

class ACHTransRecord {
    public $SSS;
    public $LocID;
    public $FrontEndTrace;
    public $OriginatorName;
    public $TransactionCode;
    public $CustTransType;
    public $CustomerID;
    public $CustomerName;
    public $CustomerRoutingNo;
    public $CustomerAcctNo;
    public $CustomerAcctType;
    public $TransAmount;
    public $CheckOrCustID;
    public $CheckOrTransDate;
    public $EffectiveDate;
    public $Memo;
    public $OpCode;
    public $AccountSet;
}

class ACHFile {
    public $SSS;
    public $LocID;
    public $ACHFileName;
    public $TotalNumRecords;
    public $TotalDebitRecords;
    public $TotalDebitAmount;
    public $TotalCreditRecords;
    public $TotalCreditAmount;
    public $ACHRecords;
}

//CompanyInfo
$mycompanyinfo = new CompanyInfo;
$mycompanyinfo -> SSS = "TST";
$mycompanyinfo -> LocID = "9505";
$mycompanyinfo -> Company = "THAT COMPANY";
$mycompanyinfo -> CompanyKey = "RICO";


//sample ACHTransRecord 1,2 & 3  (2 debits and 1 credit)
//1
$myachtransrecord1 = new ACHTransRecord;
$myachtransrecord1 -> SSS = "TST"; //T$$ assigned
$myachtransrecord1 -> LocID = "9505"; //T$$ assigned
$myachtransrecord1 -> FrontEndTrace = "W3000123456"; //impt!!!Needs a unique value each record per LocID
$myachtransrecord1 -> OriginatorName = "MYCOMPANY";
$myachtransrecord1 -> TransactionCode = "PPD"; //PPD, CCD, WEB, TEL, RCK, etc.
$myachtransrecord1 -> CustTransType = "D";  //D for debit, C for credit
$myachtransrecord1 -> CustomerID = "CustID12235";
$myachtransrecord1 -> CustomerName = "DOE, JOHNNY";
$myachtransrecord1 -> CustomerRoutingNo = "987654320";
$myachtransrecord1 -> CustomerAcctNo = "04038883674";
$myachtransrecord1 -> CustomerAcctType = "C"; //C forchecking, S for Savings
$myachtransrecord1 -> TransAmount = 100.25;
$myachtransrecord1 -> CheckOrCustID = "9166388811";
$myachtransrecord1 -> CheckOrTransDate = '2010-03-16'; //include leading zero for mm and dd e.g. 01 for Jan
$myachtransrecord1 -> EffectiveDate = '2010-03-16';    //include leading zero for mm and dd e.g. 01 for Jan
$myachtransrecord1 -> Memo = "Payment";
$myachtransrecord1 -> OpCode = "S"; //S for Single, R for Recurring
$myachtransrecord1 -> AccountSet = "1"; //T$$ assigned

//2
$myachtransrecord2 = new ACHTransRecord;
$myachtransrecord2 -> SSS = "TST"; //T$$ assigned
$myachtransrecord2 -> LocID = "9505"; //T$$ assigned
$myachtransrecord2 -> FrontEndTrace = "W3000123457"; //impt!!!Needs a unique value each record per LocID
$myachtransrecord2 -> OriginatorName = "MYCOMPANY";
$myachtransrecord2 -> TransactionCode = "PPD"; //PPD, CCD, WEB, TEL, RCK, etc.
$myachtransrecord2 -> CustTransType = "D";  //D for debit, C for credit
$myachtransrecord2 -> CustomerID = "CustID12235";
```

```
$myachtransrecord2 -> CustomerName = "DOE, JANET";
$myachtransrecord2 -> CustomerRoutingNo = "987654320";
$myachtransrecord2 -> CustomerAcctNo = "0000388836291";
$myachtransrecord2 -> CustomerAcctType = "C"; //C forchecking, S for Savings
$myachtransrecord2 -> TransAmount = 500.25;
$myachtransrecord2 -> CheckOrCustID = "9166388811";
$myachtransrecord2 -> CheckOrTransDate = '2010-03-16';  //include leading zero for mm and dd e.g. 01 for Jan
$myachtransrecord2 -> EffectiveDate = '2010-03-16';     //include leading zero for mm and dd e.g. 01 for Jan
$myachtransrecord2 -> Memo = "Payment";
$myachtransrecord2 -> OpCode = "S"; //S for Single, R for Recurring
$myachtransrecord2 -> AccountSet = "1"; //T$$ assigned

//3
$myachtransrecord3 = new ACHTransRecord;
$myachtransrecord3 -> SSS = "TST"; //T$$ assigned
$myachtransrecord3 -> LocID = "9505"; //T$$ assigned
$myachtransrecord3 -> FrontEndTrace = "W1000123451"; //impt!!!Needs a unique value each record per LocID
$myachtransrecord3 -> OriginatorName = "MYCOMPANY";
$myachtransrecord3 -> TransactionCode = "PPD"; //PPD, CCD, WEB, TEL, RCK, etc.
$myachtransrecord3 -> CustTransType = "C";  //D for debit, C for credit
$myachtransrecord3 -> CustomerID = "CustID12235";
$myachtransrecord3 -> CustomerName = "SMITH, JOE";
$myachtransrecord3 -> CustomerRoutingNo = "987654320";
$myachtransrecord3 -> CustomerAcctNo = "5500038883005";
$myachtransrecord3 -> CustomerAcctType = "C"; //C forchecking, S for Savings
$myachtransrecord3 -> TransAmount = 350.25;
$myachtransrecord3 -> CheckOrCustID = "9166388811";
$myachtransrecord3 -> CheckOrTransDate = '2010-03-16';  //include leading zero for mm and dd e.g. 01 for Jan
$myachtransrecord3 -> EffectiveDate = '2010-03-16';     //include leading zero for mm and dd e.g. 01 for Jan
$myachtransrecord3 -> Memo = "Payment";
$myachtransrecord3 -> OpCode = "S"; //S for Single, R for Recurring
$myachtransrecord3 -> AccountSet = "1"; //T$$ assigned

//ACHFile
$myachfile = new ACHFile;
$myachfile -> SSS = "TST";
$myachfile -> LocID = "9505";
$myachfile -> ACHFileName = "";  //leave blank to automatically name the file
$myachfile -> TotalNumRecords = 3;
$myachfile -> TotalDebitRecords = 2;
$myachfile -> TotalDebitAmount = 600.50;
$myachfile -> TotalCreditRecords = 1;
$myachfile -> TotalCreditAmount = 350.25;
//$myachfile -> ACHRecords = array($myachtransrecord1,$myachtransrecord2,$myachtransrecord3);  //OR
$myachfile -> ACHRecords[] = $myachtransrecord1;
$myachfile -> ACHRecords[] = $myachtransrecord2;
$myachfile -> ACHRecords[] = $myachtransrecord3;


//SOAP Call - test server
$myclient = new SoapClient("http://tstsvr.achworks.com/dnet/achws.asmx?WSDL");
$myresult = $myclient->SendACHTransBatch(array("InpCompanyInfo"=>$mycompanyinfo,"InpACHFile"=>$myachfile))-
                >SendACHTransBatchResult;

//print status and details
```

```
print($myresult->Status . ", "  . $myresult->Details . "<br>");

//print errors if there is any
foreach ($myresult->Errors->string as $myerror) {
print($myerror . "<br>");
}
?>
```

getachreturns.php

```
<?php
//ACHWORKS-SOAP Ver3.0 GetACHReturns (gets unretrieved ach return and settlement records)
//3.25.2010 - rico pamplona, rpamplonaATachworksDOTcom
//company info
class CompanyInfo {
    public $SSS;
    public $LocID;
    public $Company;
    public $CompanyKey;
}


//CompanyInfo
$mycompanyinfo = new CompanyInfo;
$mycompanyinfo -> SSS = "TST";
$mycompanyinfo -> LocID = "9505";
$mycompanyinfo -> Company = "THAT COMPANY";
$mycompanyinfo -> CompanyKey = "RICO";


//SOAP call - test server
$myclient = new SoapClient("http://tstsvr.achworks.com/dnet/achws.asmx?WSDL");
$myresult = $myclient->GetACHReturns(array("InpCompanyInfo"=>$mycompanyinfo))->GetACHReturnsResult;

//print status and details
print($myresult->Status . ", " . $myresult->Details . "<br>");

//print ACHReturnRecords if there is any
foreach ($myresult->ACHReturnRecords->ACHReturnRecord as $myACHReturnRecord) {
print("FrontEndTrace:" . $myACHReturnRecord->FrontEndTrace . ", EffectiveDate:" . $myACHReturnRecord->EffectiveDate . ",
                Name:" . $myACHReturnRecord->CustomerName . ", Amount:" . $myACHReturnRecord->TransAmount . ",
                ResponseCode:" . $myACHReturnRecord->ResponseCode . ", ActionDetail:" . $myACHReturnRecord->ActionDetail .
                "<br>");
}
?>
```

getachreturnshist.php

```
<?php
//ACHWORKS-SOAP Ver3.0 GetACHReturnsHist (gets ach return and settlement records at specified date range)
//3.25.2010 - rico pamplona, rpamplonaATachworksDOTcom
//company info
class CompanyInfo {
    public $SSS;
    public $LocID;
    public $Company;
    public $CompanyKey;
}


//CompanyInfo
$mycompanyinfo = new CompanyInfo;
$mycompanyinfo -> SSS = "TST";
$mycompanyinfo -> LocID = "9505";
$mycompanyinfo -> Company = "THAT COMPANY";
$mycompanyinfo -> CompanyKey = "RICO";

$myDateFrom = '2010-01-01'; //include leading zero for mm and dd e.g. 01 for Jan
$myDateTo = '2010-03-20';   //include leading zero for mm and dd e.g. 01 for Jan


//SOAP call - test server
$myclient = new SoapClient("http://tstsvr.achworks.com/dnet/achws.asmx?WSDL");
$myresult = $myclient->GetACHReturnsHist(array("InpCompanyInfo"=>$mycompanyinfo, "ReturnDateFrom"=>$myDateFrom,
                    "ReturnDateTo"=>$myDateTo))->GetACHReturnsHistResult;

//print status and details
print($myresult->Status . ", " . $myresult->Details . "<br>");

//print ACHReturnRecords if there is any
foreach ($myresult->ACHReturnRecords->ACHReturnRecord as $myACHReturnRecord) {
print("FrontEndTrace:" . $myACHReturnRecord->FrontEndTrace . ", EffectiveDate:" . $myACHReturnRecord->EffectiveDate . ",
                    Name:" . $myACHReturnRecord->CustomerName . ", Amount:" . $myACHReturnRecord->TransAmount . ",
                    ResponseCode:" . $myACHReturnRecord->ResponseCode . ", ActionDetail:" . $myACHReturnRecord->ActionDetail .
                    "<br>");
}
?>
```

getresultfile.php

```
<?php
//ACHWORKS-SOAP Ver3.0 GetResultFile (gets send transaction results/responses including errors at specified date range)
//3.25.2010 - rico pamplona, rpamplonaATachworksDOTcom
//company info
class CompanyInfo {
    public $SSS;
    public $LocID;
    public $Company;
    public $CompanyKey;
}


//CompanyInfo
$mycompanyinfo = new CompanyInfo;
$mycompanyinfo -> SSS = "TST";
$mycompanyinfo -> LocID = "9505";
$mycompanyinfo -> Company = "THAT COMPANY";
```

```
$mycompanyinfo -> CompanyKey = "RICO";

$myDateFrom = '2010-03-25'; //include leading zero for mm and dd e.g. 01 for Jan
$myDateTo = '2010-03-26';   //include leading zero for mm and dd e.g. 01 for Jan


//SOAP call - test server
$myclient = new SoapClient("http://tstsvr.achworks.com/dnet/achws.asmx?WSDL");
$myresult = $myclient->GetResultFile(array("InpCompanyInfo"=>$mycompanyinfo, "ResultDateFrom"=>$myDateFrom,
                    "ResultDateTo"=>$myDateTo))->GetResultFileResult;

//print status and details
print($myresult->Status . ", " . $myresult->Details . "<br><br>");

//print TransResults if there is any
print("PAST CONNECTION AND TRANSACTION RESULTS:<br>");
foreach ($myresult->TransResults->TransResult as $myTransResult) {
print("DateTime:" . $myTransResult->CallDateTime . ", Method:" . $myTransResult->CallMethod . ", Status:" . $myTransResult->Status
                    . ", FileName:" . $myTransResult->FileName  . "<br>");
   foreach ($myTransResult->Errors->string as $myError) {
   print("&nbsp&nbsp&nbsp Error=>" . $myError . "<br>");
   }
}
?>
```

geterrorfile.php

```
<?php
//ACHWORKS-SOAP Ver3.0 GetErrorFile (gets send transaction errors (only) at specified date range)
//3.25.2010 - rico pamplona, rpamplonaATachworksDOTcom
//company info
class CompanyInfo {
    public $SSS;
    public $LocID;
    public $Company;
    public $CompanyKey;
}


//CompanyInfo
$mycompanyinfo = new CompanyInfo;
$mycompanyinfo -> SSS = "TST";
$mycompanyinfo -> LocID = "9505";
$mycompanyinfo -> Company = "THAT COMPANY";
$mycompanyinfo -> CompanyKey = "RICO";

$myDateFrom = '2010-03-25'; //include leading zero for mm and dd e.g. 01 for Jan
$myDateTo = '2010-03-26';   //include leading zero for mm and dd e.g. 01 for Jan


//SOAP call - test server
$myclient = new SoapClient("http://tstsvr.achworks.com/dnet/achws.asmx?WSDL");
$myresult = $myclient->GetErrorFile(array("InpCompanyInfo"=>$mycompanyinfo, "ErrorDateFrom"=>$myDateFrom,
                    "ErrorDateTo"=>$myDateTo))->GetErrorFileResult;
```

```
//print status and details
print($myresult->Status . ", " . $myresult->Details . "<br><br>");

//print ErrorRecords if there is any
print("PAST CONNECTION AND TRANSACTION ERRORS:<br>");
foreach ($myresult->ErrorRecords->ErrorRecord as $myErrorRecord) {
print("DateTime:" . $myErrorRecord->CallDateTime . ", Method:" . $myErrorRecord->CallMethod . ", Status:" . $myErrorRecord-
                    >Status . ", FileName:" . $myErrorRecord->FileName . ", No of Errors: " . $myErrorRecord->TotalNumErrors  . "<br>");
   foreach ($myErrorRecord->Errors->string as $myError) {
   print("&nbsp&nbsp&nbsp Error=>" . $myError . "<br>");
   }
}
?>
```

A2.2 Visual Basic (VB.NET/ASP.NET)

connectioncheck

```
Sub Button1Click(sender As Object, e As EventArgs)
  'named nsACHWorksWS as the namespace when adding web reference
  Dim myACHWS As nsACHWorksWS.ACHWorksWS
  Dim myCompanyInfo As nsACHWorksWS.CompanyInfo

  'instances
  myACHWS = New nsACHWorksWS.ACHWorksWS
  myCompanyInfo = New nsACHWorksWS.CompanyInfo

  'CompanyInfo
  myCompanyInfo.SSS="TST"
  myCompanyInfo.LocID="9505"
  myCompanyInfo.Company="THAT COMPANY"
  myCompanyInfo.CompanyKey="RICO"

  'call ConnectionCheck and assign result to textBox1
  textBox1.Text=myACHWS.ConnectionCheck(myCompanyInfo)
End Sub
```

sendachtrans

```
Sub Button2Click(sender As Object, e As EventArgs)
  'named nsACHWorksWS as the namespace when adding web reference
  Dim myACHWS As nsACHWorksWS.ACHWorksWS
  Dim myCompanyInfo As nsACHWorksWS.CompanyInfo
  Dim myACHTransRecord As nsACHWorksWS.ACHTransRecord
  Dim myACHTransResult as nsACHWorksWS.TransResult


  'instances
  myACHWS = New nsACHWorksWS.ACHWorksWS
  myCompanyInfo = New nsACHWorksWS.CompanyInfo
  myACHTransRecord = New nsACHWorksWS.ACHTransRecord
  myACHTransResult = New nsACHWorksWS.TransResult


  'CompanyInfo
  myCompanyInfo.SSS="TST"
  myCompanyInfo.LocID="9505"
  myCompanyInfo.Company="THAT COMPANY"
  myCompanyInfo.CompanyKey="RICO"


  'ACHTransRecord
  myACHTransRecord.SSS="TST" 'T$$ assigned
  myACHTransRecord.LocID="9505" 'T$$ assigned
  myACHTransRecord.FrontEndTrace="VVB-0003" 'impt!!! needs a unique value each record per LocID
  myACHTransRecord.OriginatorName="MYCOMPANY"
  myACHTransRecord.TransactionCode="PPD" 'PPD,CCD,WEB,TEL,RCK, etc
  myACHTransRecord.CustTransType="D" 'D for debit, C for credit
  myACHTransRecord.CustomerID="CustID0001"
  myACHTransRecord.CustomerName="DOE, JOHN" 'Lastname, FirstName - all caps
  myACHTransRecord.CustomerRoutingNo="987654320" '9-digit Bank Routing No
  myACHTransRecord.CustomerAcctNo="034332222" 'Bank Account No
  myACHTransRecord.CustomerAcctType="C" 'C for checking, S for savings
  myACHTransRecord.TransAmount=100.75
  myACHTransRecord.CheckOrCustID="9166388811"
  myACHTransRecord.CheckOrTransDate="2010-03-24"
  myACHTransRecord.EffectiveDate="2010-03-24"
  myACHTransRecord.Memo="FirstPay"
  myACHTransRecord.OpCode="S" 'S for single, R for recurring
  myACHTransRecord.AccountSet="1" 'T$$ assigned


  'call SendACHTrans method
  myACHTransResult=myACHWS.SendACHTrans(myCompanyInfo,myACHTransRecord)


  'assign portion of result to textBox1 (e.g. Status and Details)
  textBox1.Text=myACHTransResult.Status + ", " + myACHTransResult.Details


  'print errors to listBox if there is any
  listBox1.Items.Clear
  If myACHTransResult.TotalNumErrors>0 Then
    For i=0 To myACHTransResult.TotalNumErrors-1
      listBox1.Items.Add(myACHTransResult.Errors(i).ToString)
    Next i
  End If
End Sub
```

sendachtransbatch

```
Sub Button3Click(sender As Object, e As EventArgs)
  'named nsACHWorksWS as the namespace when adding web reference
  Dim myACHWS As nsACHWorksWS.ACHWorksWS
  Dim myCompanyInfo As nsACHWorksWS.CompanyInfo
  Dim myACHTransRecord1,myACHTransRecord2,myACHTransRecord3 As nsACHWorksWS.ACHTransRecord
  Dim myACHFile As nsACHWorksWS.ACHFile
  Dim myACHTransResult As nsACHWorksWS.TransResult


  'instances
  myACHWS = New nsACHWorksWS.ACHWorksWS
  myCompanyInfo = New nsACHWorksWS.CompanyInfo
  myACHTransRecord1 = New nsACHWorksWS.ACHTransRecord
  myACHTransRecord2 = New nsACHWorksWS.ACHTransRecord
  myACHTransRecord3 = New nsACHWorksWS.ACHTransRecord
  myACHFile = new nsACHWorksWS.ACHFile
  myACHTransResult = New nsACHWorksWS.TransResult
```

```
'CompanyInfo
myCompanyInfo.SSS="TST”
myCompanyInfo.LocID="9505"
myCompanyInfo.Company="THAT COMPANY"
myCompanyInfo.CompanyKey="RICO"


'ACHFile
myACHFile.SSS="TST"
myACHFile.LocID="9505"
myACHFile.ACHFileName="" 'leave blank to automatically name the file
myACHFile.TotalNumRecords=3
myACHFile.TotalDebitRecords=2
myACHFile.TotalDebitAmount=600.50
myACHFile.TotalCreditRecords=1
myACHFile.TotalCreditAmount=350.25
myACHFile.ACHRecords = new nsACHWorksWS.ACHTransRecord(2){} '(2) is 3-1


'assign records to file
'ACHTransRecord (1, 2 & 3)
'1st record
myACHTransRecord1.SSS="TST" 'T$$ assigned
myACHTransRecord1.LocID="9505" 'T$$ assigned
myACHTransRecord1.FrontEndTrace="VVB-0003" 'impt!!! needs a unique value each record per LocID
myACHTransRecord1.OriginatorName="MYCOMPANY"
myACHTransRecord1.TransactionCode="PPD" 'PPD,CCD,WEB,TEL,RCK, etc
myACHTransRecord1.CustTransType="D" 'D for debit, C for credit
myACHTransRecord1.CustomerID="CustID0001"
myACHTransRecord1.CustomerName="DOE, JOHN" 'Lastname, FirstName - all caps
myACHTransRecord1.CustomerRoutingNo="987654320" '9-digit Bank Routing No
myACHTransRecord1.CustomerAcctNo="034332222" 'Bank Account No
myACHTransRecord1.CustomerAcctType="C" 'C for checking, S for savings
myACHTransRecord1.TransAmount=100.25
myACHTransRecord1.CheckOrCustID="9166388811"
myACHTransRecord1.CheckOrTransDate="2010-03-24"
myACHTransRecord1.EffectiveDate="2010-03-24"
myACHTransRecord1.Memo="FirstPay"
myACHTransRecord1.OpCode="S" 'S for single, R for recurring
myACHTransRecord1.AccountSet="1" 'T$$ assigned
myACHFile.ACHRecords(0)=new nsACHWorksWS.ACHTransRecord  'instance
myACHFile.ACHRecords(0)=myACHTransRecord1  'instance


'2nd record
myACHTransRecord2.SSS="TST" 'T$$ assigned
myACHTransRecord2.LocID="9505" 'T$$ assigned
myACHTransRecord2.FrontEndTrace="VVB-0004" 'impt!!! needs a unique value each record per LocID
myACHTransRecord2.OriginatorName="MYCOMPANY"
myACHTransRecord2.TransactionCode="PPD" 'PPD,CCD,WEB,TEL,RCK, etc
myACHTransRecord2.CustTransType="D" 'D for debit, C for credit
```

```
myACHTransRecord2.CustomerID="CustID0002"
myACHTransRecord2.CustomerName="DOE, JANE" 'Lastname, FirstName - all caps
myACHTransRecord2.CustomerRoutingNo="987654320" '9-digit Bank Routing No
myACHTransRecord2.CustomerAcctNo="111034332222" 'Bank Account No
myACHTransRecord2.CustomerAcctType="C" 'C for checking, S for savings
myACHTransRecord2.TransAmount=500.25
myACHTransRecord2.CheckOrCustID="9166388811"
myACHTransRecord2.CheckOrTransDate="2010-03-24"
myACHTransRecord2.EffectiveDate="2010-03-24"
myACHTransRecord2.Memo="FirstPay"
myACHTransRecord2.OpCode="S" 'S for single, R for recurring
myACHTransRecord2.AccountSet="1" 'T$$ assigned
myACHFile.ACHRecords(1)=new nsACHWorksWS.ACHTransRecord 'instance
myACHFile.ACHRecords(1)=myACHTransRecord2


'3rd record
myACHTransRecord3.SSS="TST" 'T$$ assigned
myACHTransRecord3.LocID="9505" 'T$$ assigned
myACHTransRecord3.FrontEndTrace="WVB-0005" 'impt!!! needs a unique value each record per LocID
myACHTransRecord3.OriginatorName="MYCOMPANY"
myACHTransRecord3.TransactionCode="PPD" 'PPD,CCD,WEB,TEL,RCK, etc
myACHTransRecord3.CustTransType="C" 'credit
myACHTransRecord3.CustomerID="CustID0003"
myACHTransRecord3.CustomerName="SMITH, JOE" 'Lastname, FirstName - all caps
myACHTransRecord3.CustomerRoutingNo="987654320" '9-digit Bank Routing No
myACHTransRecord3.CustomerAcctNo="111034332222" 'Bank Account No
myACHTransRecord3.CustomerAcctType="C" 'C for checking, S for savings
myACHTransRecord3.TransAmount=350.25
myACHTransRecord3.CheckOrCustID="9166388811"
myACHTransRecord3.CheckOrTransDate="2010-03-24"
myACHTransRecord3.EffectiveDate="2010-03-24"
myACHTransRecord3.Memo="FirstPay"
myACHTransRecord3.OpCode="S" 'S for single, R for recurring
myACHTransRecord3.AccountSet="1" 'T$$ assigned
myACHFile.ACHRecords(2)=new nsACHWorksWS.ACHTransRecord 'instance
myACHFile.ACHRecords(2)=myACHTransRecord3


'call SendACHTransBatch method
myACHTransResult=myACHWS.SendACHTransBatch(myCompanyInfo,myACHFile)


'assign portion of result to textBox1 (e.g. Status and Details)
textBox1.Text=myACHTransResult.Status + ", " + myACHTransResult.Details


'print errors to listBox if there is any
listBox1.Items.Clear
If myACHTransResult.TotalNumErrors>0 Then
  For i=0 To myACHTransResult.TotalNumErrors-1
    listBox1.Items.Add(myACHTransResult.Errors(i).ToString)
  Next i
  End If
End Sub
```

getachreturns

```
Sub Button4Click(sender As Object, e As EventArgs)
  Dim myACHWS As nsACHWorksWS.ACHWorksWS
  Dim myCompanyInfo As nsACHWorksWS.CompanyInfo
  Dim myACHReturns As nsACHWorksWS.ACHReturns


  'instances
  myACHWS = New nsACHWorksWS.ACHWorksWS
  myCompanyInfo = New nsACHWorksWS.CompanyInfo
  myACHReturns = New nsACHWorksWS.ACHReturns


  'CompanyInfo
  myCompanyInfo.SSS="TST"
  myCompanyInfo.LocID="9505"
  myCompanyInfo.Company="THAT COMPANY"
  myCompanyInfo.CompanyKey="RICO"


  'call GetACHReturns method
  myACHReturns = myACHWS.GetACHReturns(myCompanyInfo)
  textBox1.Text=myACHReturns.Status + ", " + myACHReturns.Details


  'list returns if there are any
  listBox1.Items.Clear
  If myACHReturns.TotalNumRecords>0 Then
    For i=0 To myACHReturns.TotalNumRecords-1
      listBox1.Items.Add("FrontEndTrace:" + myACHReturns.ACHReturnRecords(i).FrontEndTrace + _
                         ", Date:" + myACHReturns.ACHReturnRecords(i).EffectiveDate.ToString + _
                         ", Name:" + myACHReturns.ACHReturnRecords(i).CustomerName + _
                         ", Amount:" + myACHReturns.ACHReturnRecords(i).TransAmount.ToString + _
                         ", ResponseCode:" + myACHReturns.ACHReturnRecords(i).ResponseCode + _
                         ", ActionDetail:" + myACHReturns.ACHReturnRecords(i).ActionDetail)
    Next i
  End If
End Sub
```

getachreturnshist

```
Sub Button5Click(sender As Object, e As EventArgs)
  Dim myACHWS As nsACHWorksWS.ACHWorksWS
  Dim myCompanyInfo As nsACHWorksWS.CompanyInfo
  Dim myACHReturns As nsACHWorksWS.ACHReturns
  Dim myDateFrom,myDateTo as Date
  'instances
  myACHWS = New nsACHWorksWS.ACHWorksWS
  myCompanyInfo = New nsACHWorksWS.CompanyInfo
  myACHReturns = New nsACHWorksWS.ACHReturns


  'CompanyInfo
  myCompanyInfo.SSS="TST"
  myCompanyInfo.LocID="9505"
  myCompanyInfo.Company="THAT COMPANY"
  myCompanyInfo.CompanyKey="RICO"


  'date
  myDateFrom = "2010-01-01”
  myDateTo = "2010-03-20"


  'call GetACHReturnsHist method
  myACHReturns = myACHWS.GetACHReturnsHist(myCompanyInfo,myDateFrom,myDateTo)
  textBox1.Text=myACHReturns.Status + ", " + myACHReturns.Details


  'list returns if there are any
  listBox1.Items.Clear
  If myACHReturns.TotalNumRecords>0 Then
    For i=0 To myACHReturns.TotalNumRecords-1
      listBox1.Items.Add("FrontEndTrace:" + myACHReturns.ACHReturnRecords(i).FrontEndTrace + _
                         ", Date:" + myACHReturns.ACHReturnRecords(i).EffectiveDate.ToString + _
                         ", Name:" + myACHReturns.ACHReturnRecords(i).CustomerName + _
                         ", Amount:" + myACHReturns.ACHReturnRecords(i).TransAmount.ToString + _
                         ", ResponseCode:" + myACHReturns.ACHReturnRecords(i).ResponseCode + _
                         ", ActionDetail:" + myACHReturns.ACHReturnRecords(i).ActionDetail)
    Next i
  End If
End Sub
```

getresultfile

```
Sub Button6Click(sender As Object, e As EventArgs)
  Dim myACHWS As nsACHWorksWS.ACHWorksWS
  Dim myCompanyInfo As nsACHWorksWS.CompanyInfo
  Dim myResultFile As nsACHWorksWS.ResultFile
  Dim myDateFrom,myDateTo as Date


  'instances
  myACHWS = New nsACHWorksWS.ACHWorksWS
  myCompanyInfo = New nsACHWorksWS.CompanyInfo
  myResultFile = New nsACHWorksWS.ResultFile
  'CompanyInfo
  myCompanyInfo.SSS="TST"
  myCompanyInfo.LocID="9505"
  myCompanyInfo.Company="THAT COMPANY"
  myCompanyInfo.CompanyKey="RICO"


  'date
  myDateFrom = "2010-03-25"
  myDateTo = "2010-03-26"


  'call GetResultFile method
  myResultFile = myACHWS.GetResultFile(myCompanyInfo,myDateFrom,myDateTo)


  'assign status/details to textbox
  textBox1.Text=myResultFile.Status + ", " + myResultFile.Details


  'list past connection results to listBox1
  listBox1.Items.Clear
  If myResultFile.TotalResultRecords>0 Then
    For i=0 To myResultFile.TotalResultRecords-1
      listBox1.Items.Add("DateTime:" + myResultFile.TransResults(i).CallDateTime.ToString + _
                         ", Method:" + myResultFile.TransResults(i).CallMethod + _
                         ", Status:" + myResultFile.TransResults(i).Status + _
                         ", FileName:" + myResultFile.TransResults(i).FileName)
      For j=0 To myResultFile.TransResults(i).TotalNumErrors-1
        listBox1.Items.Add(">" + myResultFile.TransResults(i).Errors(j).ToString)
      Next j
    Next i
  End If
End Sub
```

geterrorfile

```
Sub Button7Click(sender As Object, e As EventArgs)
  Dim myACHWS As nsACHWorksWS.ACHWorksWS
  Dim myCompanyInfo As nsACHWorksWS.CompanyInfo
  Dim myErrorFile As nsACHWorksWS.ErrorFile
  Dim myDateFrom,myDateTo as Date


  'instances
  myACHWS = New nsACHWorksWS.ACHWorksWS
  myCompanyInfo = New nsACHWorksWS.CompanyInfo
  myErrorFile = New nsACHWorksWS.ErrorFile
  'CompanyInfo
  myCompanyInfo.SSS="TST"
  myCompanyInfo.LocID="9505"
  myCompanyInfo.Company="THAT COMPANY"
  myCompanyInfo.CompanyKey="RICO"


  'date
  myDateFrom = "2010-03-25"
  myDateTo = "2010-03-26"


  'call GetErrorFile method
  myErrorFile = myACHWS.GetErrorFile(myCompanyInfo,myDateFrom,myDateTo)


  'assign status/details to textbox
  textBox1.Text=myErrorFile.Status + ", " + myErrorFile.Details


  'list past connection errors to listBox1
  listBox1.Items.Clear
  If myErrorFile.TotalErrorRecords>0 Then
    For i=0 To myErrorFile.TotalErrorRecords-1
      listBox1.Items.Add("DateTime:" + myErrorFile.ErrorRecords(i).CallDateTime.ToString + _
                         ", Method:" + myErrorFile.ErrorRecords(i).CallMethod + _
                         ", FileName:" + myErrorFile.ErrorRecords(i).FileName + _
                         ", No of Errors:" + myErrorFile.ErrorRecords(i).TotalNumErrors.ToString)
      For j=0 To myErrorFile.ErrorRecords(i).TotalNumErrors-1
        listBox1.Items.Add(">" + myErrorFile.ErrorRecords(i).Errors(j).ToString)
      Next j
    Next i
  End If
End Sub
```

A2.3 C# (.NET)

connectioncheck

```
void Button1Click(object sender, EventArgs e)
{
  //named nsACHWorksWS as the namespace when adding web reference
  //instances
  nsACHWorksWS.ACHWorksWS myACHWS = new nsACHWorksWS.ACHWorksWS();
  nsACHWorksWS.CompanyInfo  myCompanyInfo = new nsACHWorksWS.CompanyInfo();


  //company info
  myCompanyInfo.SSS="TST";
  myCompanyInfo.LocID="9505";
  myCompanyInfo.Company="THAT COMPANY";
  myCompanyInfo.CompanyKey="RICO";


  //call ConnectionCheck method
  textBox1.Text=myACHWS.ConnectionCheck(myCompanyInfo);
}
```

sendachtrans

```
void Button2Click(object sender, EventArgs e)
{
  //instances
  nsACHWorksWS.ACHWorksWS myACHWS = new nsACHWorksWS.ACHWorksWS();
  nsACHWorksWS.CompanyInfo  myCompanyInfo = new nsACHWorksWS.CompanyInfo();
  nsACHWorksWS.ACHTransRecord myACHTransRecord = new nsACHWorksWS.ACHTransRecord();
  nsACHWorksWS.TransResult myACHTransResult = new nsACHWorksWS.TransResult();


  //company info
  myCompanyInfo.SSS="TST";
  myCompanyInfo.LocID="9505";
  myCompanyInfo.Company="THAT COMPANY";
  myCompanyInfo.CompanyKey="RICO";


  //ACHTransRecord
  myACHTransRecord.SSS="TST"; //T$$ assigned
  myACHTransRecord.LocID="9505"; //T$$ assigned
  myACHTransRecord.FrontEndTrace="CC#-0003"; //impt! needs a unique value each record per LocID
  myACHTransRecord.OriginatorName="MYCOMPANY";
  myACHTransRecord.TransactionCode="PPD"; //PPD,CCD,WEB,TEL,RCK, etc.
  myACHTransRecord.CustTransType="D"; //D for debit, C for credit
  myACHTransRecord.CustomerID="CustID0001";
  myACHTransRecord.CustomerName="DOE, JOHN";  //Lastname, Firstname - all caps
  myACHTransRecord.CustomerRoutingNo="987654320"; //9-digit Bank Routing No
  myACHTransRecord.CustomerAcctNo="0002311331555"; //Bank Account No
  myACHTransRecord.CustomerAcctType="C"; //C for checking, S for savings
  myACHTransRecord.TransAmount=100.75;
  myACHTransRecord.CheckOrCustID="9166388811";
  myACHTransRecord.CheckOrTransDate=Convert.ToDateTime("2010-03-24");
  myACHTransRecord.EffectiveDate=Convert.ToDateTime("2010-03-24");
  myACHTransRecord.Memo="FirstPay";
  myACHTransRecord.OpCode="S"; //S for single, R for recurring
  myACHTransRecord.AccountSet="1";  //T$$ assigned


  //call SendACHTrans method
  myACHTransResult=myACHWS.SendACHTrans(myCompanyInfo,myACHTransRecord);


  //assign portion of result to textbox1 (e.g. status and details)
  textBox1.Text=myACHTransResult.Status + ", " + myACHTransResult.Details;


  //print errors to listBox if there is any
  listBox1.Items.Clear();
  if (myACHTransResult.TotalNumErrors>0)
  {
    for (int i=0; i<myACHTransResult.TotalNumErrors; i++)
      listBox1.Items.Add(myACHTransResult.Errors[i]);
  }
}
```

sendachtransbatch

```
void Button3Click(object sender, EventArgs e)
{
  //instances
  nsACHWorksWS.ACHWorksWS myACHWS = new nsACHWorksWS.ACHWorksWS();
  nsACHWorksWS.CompanyInfo  myCompanyInfo = new nsACHWorksWS.CompanyInfo();
  nsACHWorksWS.ACHTransRecord myACHTransRecord1 = new nsACHWorksWS.ACHTransRecord();
  nsACHWorksWS.ACHTransRecord myACHTransRecord2 = new nsACHWorksWS.ACHTransRecord();
  nsACHWorksWS.ACHTransRecord myACHTransRecord3 = new nsACHWorksWS.ACHTransRecord();
  nsACHWorksWS.ACHFile myACHFile = new nsACHWorksWS.ACHFile();
  nsACHWorksWS.TransResult myACHTransResult = new nsACHWorksWS.TransResult();


  //company info
  myCompanyInfo.SSS="TST";
  myCompanyInfo.LocID="9505";
  myCompanyInfo.Company="THAT COMPANY";
  myCompanyInfo.CompanyKey="RICO";


  //ACHFile
  myACHFile.SSS="TST";
  myACHFile.LocID="9505";
  myACHFile.ACHFileName=""; //leave blank to automatically name the file
  myACHFile.TotalNumRecords=3;
  myACHFile.TotalDebitRecords=2;
  myACHFile.TotalDebitAmount=600.50;
  myACHFile.TotalCreditRecords=1;
  myACHFile.TotalCreditAmount=350.25;
  myACHFile.ACHRecords=new nsACHWorksWS.ACHTransRecord[3];


  //ACHTransRecord(s)
  //Record 1
  myACHTransRecord1.SSS="TST"; //T$$ assigned
  myACHTransRecord1.LocID="9505"; //T$$ assigned
  myACHTransRecord1.FrontEndTrace="CC#-00031"; //impt! needs a unique value each record per LocID
  myACHTransRecord1.OriginatorName="MYCOMPANY";
  myACHTransRecord1.TransactionCode="PPD"; //PPD,CCD,WEB,TEL,RCK, etc.
  myACHTransRecord1.CustTransType="D"; //D for debit, C for credit
  myACHTransRecord1.CustomerID="CustID0001";
  myACHTransRecord1.CustomerName="DOE, JOHN";  //Lastname, Firstname - all caps
  myACHTransRecord1.CustomerRoutingNo="987654320"; //9-digit Bank Routing No
  myACHTransRecord1.CustomerAcctNo="0002311331555"; //Bank Account No
  myACHTransRecord1.CustomerAcctType="C"; //C for checking, S for savings
  myACHTransRecord1.TransAmount=100.25;
  myACHTransRecord1.CheckOrCustID="9166388811";
  myACHTransRecord1.CheckOrTransDate=Convert.ToDateTime("2010-03-24");
  myACHTransRecord1.EffectiveDate=Convert.ToDateTime("2010-03-24");
  myACHTransRecord1.Memo="FirstPay";
  myACHTransRecord1.OpCode="S"; //S for single, R for recurring
  myACHTransRecord1.AccountSet="1";  //T$$ assigned
  myACHFile.ACHRecords[0]=new nsACHWorksWS.ACHTransRecord();
  myACHFile.ACHRecords[0]=myACHTransRecord1;


  //Record 2
  myACHTransRecord2.SSS="TST"; //T$$ assigned
  myACHTransRecord2.LocID="9505"; //T$$ assigned
  myACHTransRecord2.FrontEndTrace="CC#-00032"; //impt! needs a unique value each record per LocID
  myACHTransRecord2.OriginatorName="MYCOMPANY";
  myACHTransRecord2.TransactionCode="PPD"; //PPD,CCD,WEB,TEL,RCK, etc.
  myACHTransRecord2.CustTransType="D"; //D for debit, C for credit
  myACHTransRecord2.CustomerID="CustID0002";
  myACHTransRecord2.CustomerName="DOE, JANE";  //Lastname, Firstname - all caps
  myACHTransRecord2.CustomerRoutingNo="987654320"; //9-digit Bank Routing No
  myACHTransRecord2.CustomerAcctNo="400231133002"; //Bank Account No
  myACHTransRecord2.CustomerAcctType="C"; //C for checking, S for savings
  myACHTransRecord2.TransAmount=500.25;
  myACHTransRecord2.CheckOrCustID="9166388811";
  myACHTransRecord2.CheckOrTransDate=Convert.ToDateTime("2010-03-24");
  myACHTransRecord2.EffectiveDate=Convert.ToDateTime("2010-03-24");
  myACHTransRecord2.Memo="FirstPay";
  myACHTransRecord2.OpCode="S"; //S for single, R for recurring
  myACHTransRecord2.AccountSet="1";  //T$$ assigned
  myACHFile.ACHRecords[1]=new nsACHWorksWS.ACHTransRecord();
```

```
  myACHFile.ACHRecords[1]=myACHTransRecord2;


  //Record 3
  myACHTransRecord3.SSS="TST"; //T$$ assigned
  myACHTransRecord3.LocID="9505"; //T$$ assigned
  myACHTransRecord3.FrontEndTrace="CC#-00033"; //impt! needs a unique value each record per LocID
  myACHTransRecord3.OriginatorName="MYCOMPANY";
  myACHTransRecord3.TransactionCode="PPD"; //PPD,CCD,WEB,TEL,RCK, etc.
  myACHTransRecord3.CustTransType="C"; //D for debit, C for credit
  myACHTransRecord3.CustomerID="CustID0003";
  myACHTransRecord3.CustomerName="SMITH, JOE";  //Lastname, Firstname - all caps
  myACHTransRecord3.CustomerRoutingNo="987654320"; //9-digit Bank Routing No
  myACHTransRecord3.CustomerAcctNo="90200231552"; //Bank Account No
  myACHTransRecord3.CustomerAcctType="C"; //C for checking, S for savings
  myACHTransRecord3.TransAmount=350.25;
  myACHTransRecord3.CheckOrCustID="9166388811";
  myACHTransRecord3.CheckOrTransDate=Convert.ToDateTime("2010-03-24");
  myACHTransRecord3.EffectiveDate=Convert.ToDateTime("2010-03-24");
  myACHTransRecord3.Memo="FirstPay";
  myACHTransRecord3.OpCode="S"; //S for single, R for recurring
  myACHTransRecord3.AccountSet="1";  //T$$ assigned
  myACHFile.ACHRecords[2]=new nsACHWorksWS.ACHTransRecord();
  myACHFile.ACHRecords[2]=myACHTransRecord3;


  //call SendACHTransBatch method
  myACHTransResult=myACHWS.SendACHTransBatch(myCompanyInfo,myACHFile);


  //assign portion of result to textbox1 (e.g. status and details)
  textBox1.Text=myACHTransResult.Status + ", " + myACHTransResult.Details;


  //print errors to listBox if there is any
  listBox1.Items.Clear();
  if (myACHTransResult.TotalNumErrors>0)
  {
    for (int i=0; i<myACHTransResult.TotalNumErrors; i++)
      listBox1.Items.Add(myACHTransResult.Errors[i]);
  }
}
```

getachreturns

```
void Button4Click(object sender, EventArgs e)
{
  //instances
  nsACHWorksWS.ACHWorksWS myACHWS = new nsACHWorksWS.ACHWorksWS();
  nsACHWorksWS.CompanyInfo  myCompanyInfo = new nsACHWorksWS.CompanyInfo();
  nsACHWorksWS.ACHReturns myACHReturns = new nsACHWorksWS.ACHReturns();
  //company info
  myCompanyInfo.SSS="TST";
  myCompanyInfo.LocID="9505";
  myCompanyInfo.Company="THAT COMPANY";
  myCompanyInfo.CompanyKey="RICO";


  //call GetACHReturns method
  myACHReturns=myACHWS.GetACHReturns(myCompanyInfo);
  textBox1.Text=myACHReturns.Status + ", " + myACHReturns.Details;


  //list returns if there is any
  listBox1.Items.Clear();
  if (myACHReturns.TotalNumRecords>0)
  {
    for (int i=0; i<myACHReturns.TotalNumRecords; i++)
      listBox1.Items.Add("FrontEndTrace:" + myACHReturns.ACHReturnRecords[i].FrontEndTrace +
                        ", EffectiveDate:" + myACHReturns.ACHReturnRecords[i].EffectiveDate.ToString() +
                        ", Name:" + myACHReturns.ACHReturnRecords[i].CustomerName +
                        ", Amount:" + myACHReturns.ACHReturnRecords[i].TransAmount.ToString() +
                        ", ResponseCode:" + myACHReturns.ACHReturnRecords[i].ResponseCode +
                        ", ActionDetail:" + myACHReturns.ACHReturnRecords[i].ActionDetail.Trim() +
                        ", ActionDate:" + myACHReturns.ACHReturnRecords[i].ActionDate.ToString());

  }
}
```

getachreturnshist

```
void Button5Click(object sender, EventArgs e)
{
  //instances
  nsACHWorksWS.ACHWorksWS myACHWS = new nsACHWorksWS.ACHWorksWS();
  nsACHWorksWS.CompanyInfo  myCompanyInfo = new nsACHWorksWS.CompanyInfo();
  nsACHWorksWS.ACHReturns myACHReturns = new nsACHWorksWS.ACHReturns();
  DateTime myDateFrom, myDateTo;


  //company info
  myCompanyInfo.SSS="TST";
  myCompanyInfo.LocID="9505";
  myCompanyInfo.Company="THAT COMPANY";
  myCompanyInfo.CompanyKey="RICO";




  //date
  myDateFrom=Convert.ToDateTime("2010-01-01");
  myDateTo=Convert.ToDateTime("2010-03-20");
  //call GetACHReturnsHist method
  myACHReturns=myACHWS.GetACHReturnsHist(myCompanyInfo,myDateFrom,myDateTo);
  textBox1.Text=myACHReturns.Status + ", " + myACHReturns.Details;


  //list returns if there is any
  listBox1.Items.Clear();
  if (myACHReturns.TotalNumRecords>0)
  {
    for (int i=0; i<myACHReturns.TotalNumRecords; i++)
      listBox1.Items.Add("FrontEndTrace:" + myACHReturns.ACHReturnRecords[i].FrontEndTrace +
                        ", EffectiveDate:" + myACHReturns.ACHReturnRecords[i].EffectiveDate.ToString() +
                        ", Name:" + myACHReturns.ACHReturnRecords[i].CustomerName +
                        ", Amount:" + myACHReturns.ACHReturnRecords[i].TransAmount.ToString() +
                        ", ResponseCode:" + myACHReturns.ACHReturnRecords[i].ResponseCode +
                        ", ActionDetail:" + myACHReturns.ACHReturnRecords[i].ActionDetail +
                        ", ActionDate:" + myACHReturns.ACHReturnRecords[i].ActionDate.ToString());

  }
}
```

getresultfile

```
void Button6Click(object sender, EventArgs e)
{
  //instances
  nsACHWorksWS.ACHWorksWS myACHWS = new nsACHWorksWS.ACHWorksWS();
  nsACHWorksWS.CompanyInfo  myCompanyInfo = new nsACHWorksWS.CompanyInfo();
  nsACHWorksWS.ResultFile myResultFile = new nsACHWorksWS.ResultFile();
  DateTime myDateFrom, myDateTo;


  //company info
  myCompanyInfo.SSS="TST";
  myCompanyInfo.LocID="9505";
  myCompanyInfo.Company="THAT COMPANY";
  myCompanyInfo.CompanyKey="RICO";


  //date
  myDateFrom=Convert.ToDateTime("2010-03-25");
  myDateTo=Convert.ToDateTime("2010-03-26");


  //call GetResultFile method
  myResultFile=myACHWS.GetResultFile(myCompanyInfo,myDateFrom,myDateTo);
  textBox1.Text=myResultFile.Status + ", " + myResultFile.Details;
  //list past connection results
  listBox1.Items.Clear();
  if (myResultFile.TotalResultRecords>0)
  {
    for (int i=0; i<myResultFile.TotalResultRecords; i++)
    {
      listBox1.Items.Add("DateTime:" + myResultFile.TransResults[i].CallDateTime.ToString() +
                         ", Method:" + myResultFile.TransResults[i].CallMethod +
                         ", Status:" + myResultFile.TransResults[i].Status +
                         ", FileName:" + myResultFile.TransResults[i].FileName);
      for (int j=0; j<myResultFile.TransResults[i].TotalNumErrors; j++)
        listBox1.Items.Add(">" + myResultFile.TransResults[i].Errors[j]);


    }
  }
}
```

geterrorfile

```
void Button7Click(object sender, EventArgs e)
{
  //instances
  nsACHWorksWS.ACHWorksWS myACHWS = new nsACHWorksWS.ACHWorksWS();
  nsACHWorksWS.CompanyInfo  myCompanyInfo = new nsACHWorksWS.CompanyInfo();
  nsACHWorksWS.ErrorFile myErrorFile = new nsACHWorksWS.ErrorFile();
  DateTime myDateFrom, myDateTo;


  //company info
  myCompanyInfo.SSS="TST";
  myCompanyInfo.LocID="9505";
  myCompanyInfo.Company="THAT COMPANY";
  myCompanyInfo.CompanyKey="RICO";


  //date
  myDateFrom=Convert.ToDateTime("2010-03-25");
  myDateTo=Convert.ToDateTime("2010-03-26");


  //call GetErrorFile method
  myErrorFile=myACHWS.GetErrorFile(myCompanyInfo,myDateFrom,myDateTo);
  textBox1.Text=myErrorFile.Status + ", " + myErrorFile.Details;




  //list error records
  listBox1.Items.Clear();
  if (myErrorFile.TotalErrorRecords>0)
  {
    for (int i=0; i<myErrorFile.TotalErrorRecords; i++)
    {
      listBox1.Items.Add("DateTime:" + myErrorFile.ErrorRecords[i].CallDateTime.ToString() +
                         ", Method:" +  myErrorFile.ErrorRecords[i].CallMethod +
                         ", FileName:" +  myErrorFile.ErrorRecords[i].FileName +
                         ", No of Errors:" +  myErrorFile.ErrorRecords[i].TotalNumErrors.ToString());
      for (int j=0; j< myErrorFile.ErrorRecords[i].TotalNumErrors; j++)
        listBox1.Items.Add(">" +  myErrorFile.ErrorRecords[i].Errors[j]);
    }
  }
}
```

A2.4 Java (1.5/1.6)

connectioncheck

```
 private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
   try
   {
     //instances
     ACHWorksWS myACHWS = new ACHWorksWS();
     CompanyInfo myCompanyInfo = new CompanyInfo();

     //assign values for myCompanyInfo
     myCompanyInfo.sss="TST";
     myCompanyInfo.locID="9505";
     myCompanyInfo.company="THAT COMPANY";
     myCompanyInfo.companyKey="RICO";


     //put value of ConnectionCheck call to jTextField
     jTextField1.setText(myACHWS.getACHWorksWSSoap().connectionCheck(myCompanyInfo));
   }
   catch (Exception e)
   {
     jTextField1.setText("Exception caught: " + e);
   }
}
```

sendachtrans

```
 private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
  //need to 'import' the following:
  //import java.util.*;
  //import java.text.*;
  //import javax.xml.datatype.DatatypeConfigurationException;
  //import javax.xml.datatype.DatatypeFactory;
  //import javax.xml.datatype.XMLGregorianCalendar;
  //import javax.swing.DefaultListModel;

  try
  {
    //instances
    ACHWorksWS myACHWS = new ACHWorksWS();
    CompanyInfo myCompanyInfo = new CompanyInfo();
    ACHTransRecord myACHTransRecord = new ACHTransRecord();
    TransResult myACHTransResult = new TransResult();
    DatatypeFactory myDTF;


    //assign values for myCompanyInfo
    myCompanyInfo.sss="TST";
    myCompanyInfo.locID="9505";
    myCompanyInfo.company="THAT COMPANY";
    myCompanyInfo.companyKey="RICO";
    //ACHTransRecord
    myACHTransRecord.sss="TST"; //T$$ assigned
    myACHTransRecord.locID="9505"; //T$$ assigned
    myACHTransRecord.frontEndTrace="JJ-0007"; //impt! needs a unique value for each record per locid
    myACHTransRecord.originatorName="MYCOMPANY";
    myACHTransRecord.transactionCode="PPD"; //PPD,CCD,WEB,TEL,RCK, etc.
    myACHTransRecord.custTransType="D"; //D for debit, C for credit
    myACHTransRecord.customerID="CUSTID123";
    myACHTransRecord.customerName="DOE, JOHN";  //Lastname, Firstname - all caps
    myACHTransRecord.customerRoutingNo="987654320"; //9-digit Bank Routing No
    myACHTransRecord.customerAcctNo="00332358882"; //Bank Account No
    myACHTransRecord.customerAcctType="C"; //C for checking, S for savings
    myACHTransRecord.transAmount=100.75;
    myACHTransRecord.checkOrCustID="9166388811";

    try
    {
       myDTF = DatatypeFactory.newInstance();
       myACHTransRecord.checkOrTransDate=myDTF.newXMLGregorianCalendar("2010-03-24T00:00:00");
       myACHTransRecord.effectiveDate=myDTF.newXMLGregorianCalendar("2010-03-24T00:00:00");
    }
    catch (DatatypeConfigurationException e)
    {
       jTextField1.setText("Date exception caught: " + e);
    }

    myACHTransRecord.memo="FirstPay";
    myACHTransRecord.opCode="S";  //S for single, R for recurring
    myACHTransRecord.accountSet="1";  //T$$ assigned

    //call SendACHTrans method
    myACHTransResult = myACHWS.getACHWorksWSSoap().sendACHTrans(myCompanyInfo, myACHTransRecord);
    jTextField1.setText(myACHTransResult.status + ", " + myACHTransResult.details);


    //list errors if there are any
    if (myACHTransResult.totalNumErrors>0)
    {
      DefaultListModel myList = new DefaultListModel();

      jList1.setVisibleRowCount(myACHTransResult.totalNumErrors);
      for (int i=0; i<myACHTransResult.totalNumErrors; i++)
         myList.addElement(myACHTransResult.errors.string.get(i));

      jList1.setModel(myList);
     }
  }
  catch (Exception e)
  {
     jTextField1.setText("Exception caught: " + e);
  }
}
```

sendachtransbatch

```
 private void jButton3MouseClicked(java.awt.event.MouseEvent evt) {
  //need to 'import' the following:
  //import java.util.*;
  //import java.text.*;
  //import javax.xml.datatype.DatatypeConfigurationException;
  //import javax.xml.datatype.DatatypeFactory;
  //import javax.xml.datatype.XMLGregorianCalendar;
  //import javax.swing.DefaultListModel;

  try
  {
    //instances
    ACHWorksWS myACHWS = new ACHWorksWS();
    CompanyInfo myCompanyInfo = new CompanyInfo();
    ACHTransRecord myACHTransRecord1 = new ACHTransRecord();
    ACHTransRecord myACHTransRecord2 = new ACHTransRecord();
    ACHTransRecord myACHTransRecord3 = new ACHTransRecord();
    ACHFile myACHFile = new ACHFile();
    TransResult myACHTransResult = new TransResult();
    DatatypeFactory myDTF;


    //assign values for myCompanyInfo
    myCompanyInfo.sss="TST";
    myCompanyInfo.locID="9505";
    myCompanyInfo.company="THAT COMPANY";
    myCompanyInfo.companyKey="RICO";


    //myACHFile
    myACHFile.sss="TST";
    myACHFile.locID="9505";
    myACHFile.achFileName="";  //leave blank to automatically name the file
    myACHFile.totalNumRecords=3;
    myACHFile.totalDebitRecords=2;
    myACHFile.totalDebitAmount=600.50;
    myACHFile.totalCreditRecords=1;
    myACHFile.totalCreditAmount=350.25;

    //initialize records
    myACHFile.achRecords = new ArrayOfACHTransRecord();


    //ACHTransRecord(s)
    //Record 1
    myACHTransRecord1.sss="TST"; //T$$ assigned
    myACHTransRecord1.locID="9505"; //T$$ assigned
    myACHTransRecord1.frontEndTrace="JJ-00081"; //impt! needs a unique value for each record per locid
    myACHTransRecord1.originatorName="MYCOMPANY";
    myACHTransRecord1.transactionCode="PPD"; //PPD,CCD,WEB,TEL,RCK, etc.
    myACHTransRecord1.custTransType="D"; //D for debit, C for credit
    myACHTransRecord1.customerID="CUSTID123-1";
    myACHTransRecord1.customerName="DOE, JOHN";  //Lastname, Firstname - all caps
```

```
myACHTransRecord1.customerRoutingNo="987654320"; //9-digit Bank Routing No
myACHTransRecord1.customerAcctNo="00332358882"; //Bank Account No
myACHTransRecord1.customerAcctType="C"; //C for checking, S for savings
myACHTransRecord1.transAmount=100.25;
myACHTransRecord1.checkOrCustID="9166388811";

try
{
   myDTF = DatatypeFactory.newInstance();
   myACHTransRecord1.checkOrTransDate=myDTF.newXMLGregorianCalendar("2010-03-24T00:00:00");
   myACHTransRecord1.effectiveDate=myDTF.newXMLGregorianCalendar("2010-03-24T00:00:00");
}
catch (DatatypeConfigurationException e)
{
   jTextField1.setText("Date exception caught: " + e);
}

myACHTransRecord1.memo="FirstPay";
myACHTransRecord1.opCode="S";  //S for single, R for recurring
myACHTransRecord1.accountSet="1";  //T$$ assigned


//Record 2
myACHTransRecord2.sss="TST"; //T$$ assigned
myACHTransRecord2.locID="9505"; //T$$ assigned
myACHTransRecord2.frontEndTrace="JJ-00082"; //impt! needs a unique value for each record per locid
myACHTransRecord2.originatorName="MYCOMPANY";
myACHTransRecord2.transactionCode="PPD"; //PPD,CCD,WEB,TEL,RCK, etc.
myACHTransRecord2.custTransType="D"; //D for debit, C for credit
myACHTransRecord2.customerID="CUSTID123-2";
myACHTransRecord2.customerName="SMITH, JANET";  //Lastname, Firstname - all caps
myACHTransRecord2.customerRoutingNo="987654320"; //9-digit Bank Routing No
myACHTransRecord2.customerAcctNo="00332358882"; //Bank Account No
myACHTransRecord2.customerAcctType="C"; //C for checking, S for savings
myACHTransRecord2.transAmount=500.25;
myACHTransRecord2.checkOrCustID="9166388811";

try
{
   myDTF = DatatypeFactory.newInstance();
   myACHTransRecord2.checkOrTransDate=myDTF.newXMLGregorianCalendar("2010-03-24T00:00:00");
   myACHTransRecord2.effectiveDate=myDTF.newXMLGregorianCalendar("2010-03-24T00:00:00");
}
catch (DatatypeConfigurationException e)
{
   jTextField1.setText("Date exception caught: " + e);
}

myACHTransRecord2.memo="FirstPay";
myACHTransRecord2.opCode="S";  //S for single, R for recurring
myACHTransRecord2.accountSet="1";  //T$$ assigned


//Record 3
myACHTransRecord3.sss="TST"; //T$$ assigned
```

```
  myACHTransRecord3.locID="9505"; //T$$ assigned
  myACHTransRecord3.frontEndTrace="JJ-00083"; //impt! needs a unique value for each record per locid
  myACHTransRecord3.originatorName="MYCOMPANY";
  myACHTransRecord3.transactionCode="PPD"; //PPD,CCD,WEB,TEL,RCK, etc.
  myACHTransRecord3.custTransType="C"; //D for debit, C for credit
  myACHTransRecord3.customerID="CUSTID123-3";
  myACHTransRecord3.customerName="YOUNG, JOE";  //Lastname, Firstname - all caps
  myACHTransRecord3.customerRoutingNo="987654320"; //9-digit Bank Routing No
  myACHTransRecord3.customerAcctNo="00332358882"; //Bank Account No
  myACHTransRecord3.customerAcctType="C"; //C for checking, S for savings
  myACHTransRecord3.transAmount=350.25;
  myACHTransRecord3.checkOrCustID="9166388811";

  try
  {
     myDTF = DatatypeFactory.newInstance();
     myACHTransRecord3.checkOrTransDate=myDTF.newXMLGregorianCalendar("2010-03-24T00:00:00");
     myACHTransRecord3.effectiveDate=myDTF.newXMLGregorianCalendar("2010-03-24T00:00:00");
  }
  catch (DatatypeConfigurationException e)
  {
     jTextField1.setText("Date exception caught: " + e);
  }

  myACHTransRecord3.memo="FirstPay";
  myACHTransRecord3.opCode="S";  //S for single, R for recurring
  myACHTransRecord3.accountSet="1";  //T$$ assigned


  //add ACHTransRecords to ACHFile
  myACHFile.achRecords.getACHTransRecord().add(myACHTransRecord1);
  myACHFile.achRecords.getACHTransRecord().add(myACHTransRecord2);
  myACHFile.achRecords.getACHTransRecord().add(myACHTransRecord3);


  //call SendACHTransBatch method
  myACHTransResult = myACHWS.getACHWorksWSSoap().sendACHTransBatch(myCompanyInfo, myACHFile);
  jTextField1.setText(myACHTransResult.status + ", " + myACHTransResult.details);


  //list errors if there are any
  if (myACHTransResult.totalNumErrors>0)
  {
    DefaultListModel myList = new DefaultListModel();

    jList1.setVisibleRowCount(myACHTransResult.totalNumErrors);
    for (int i=0; i<myACHTransResult.totalNumErrors; i++)
       myList.addElement(myACHTransResult.errors.string.get(i));

    jList1.setModel(myList);
   }
}
catch (Exception e)
{
   jTextField1.setText("Exception caught: " + e);
```

```
    }
}
```

getachreturns

```
private void jButton4MouseClicked(java.awt.event.MouseEvent evt) {
   //need to 'import' the following:
   //import java.util.*;
   //import java.text.*;
   //import javax.swing.DefaultListModel;

   try
   {
     //instances
     ACHWorksWS myACHWS = new ACHWorksWS();
     CompanyInfo myCompanyInfo = new CompanyInfo();
     ACHReturns myACHReturns = new ACHReturns();


     //assign values for myCompanyInfo
     myCompanyInfo.sss="TST";
     myCompanyInfo.locID="9505";
     myCompanyInfo.company="THAT COMPANY";
     myCompanyInfo.companyKey="RICO";


     //call GetACHReturns method
     myACHReturns = myACHWS.getACHWorksWSSoap().getACHReturns(myCompanyInfo);
     jTextField1.setText(myACHReturns.status + ", " + myACHReturns.details);


     //list ACHReturns if there are any
     if (myACHReturns.totalNumRecords>0)
     {
       DefaultListModel myList = new DefaultListModel();

       jList1.setVisibleRowCount(myACHReturns.totalNumRecords);
       for (int i=0; i<myACHReturns.totalNumRecords; i++)
         myList.addElement("FrontEndTrace:" +
            myACHReturns.achReturnRecords.getACHReturnRecord().get(i).frontEndTrace +
               ", EffectiveDate:" +
            myACHReturns.achReturnRecords.getACHReturnRecord().get(i).effectiveDate.toString() +
               ", Name:" + myACHReturns.achReturnRecords.getACHReturnRecord().get(i).customerName +
                           ", Amount:" +
            myACHReturns.achReturnRecords.getACHReturnRecord().get(i).transAmount.toString() +
               ", ResponseCode:" +
            myACHReturns.achReturnRecords.getACHReturnRecord().get(i).responseCode +
               ", ActionDetail:" +
            myACHReturns.achReturnRecords.getACHReturnRecord().get(i).actionDetail +
                           ", ActionDate:" +
            myACHReturns.achReturnRecords.getACHReturnRecord().get(i).actionDate.toString());


       jList1.setModel(myList);
      }
   }
   catch (Exception e)
   {
      jTextField1.setText("Exception caught: " + e);
   }
    }
```

getachreturnshist

```
private void jButton5MouseClicked(java.awt.event.MouseEvent evt) {
  //need to 'import' the following:
  //import java.util.*;
  //import java.text.*;
  //import javax.xml.datatype.DatatypeConfigurationException;
  //import javax.xml.datatype.DatatypeFactory;
  //import javax.xml.datatype.XMLGregorianCalendar;
  //import javax.swing.DefaultListModel;

  try
  {
    //instances
    ACHWorksWS myACHWS = new ACHWorksWS();
    CompanyInfo myCompanyInfo = new CompanyInfo();
    ACHReturns myACHReturns = new ACHReturns();
    DatatypeFactory myDTF;


    //assign values for myCompanyInfo
    myCompanyInfo.sss="TST";
    myCompanyInfo.locID="9505";
    myCompanyInfo.company="THAT COMPANY";
    myCompanyInfo.companyKey="RICO";

    try
    {
       //instance
       myDTF = DatatypeFactory.newInstance();

       //call GetACHReturnsHist method
       myACHReturns =
            myACHWS.getACHWorksWSSoap().getACHReturnsHist(myCompanyInfo,myDTF.newXMLGregorianCalendar("2
            010-01-01T00:00:00"),myDTF.newXMLGregorianCalendar("2010-03-20T00:00:00"));
    }
    catch (DatatypeConfigurationException e)
    {
       jTextField1.setText("Date exception caught: " + e);
    }

    //assign portion of result to jTextField
    jTextField1.setText(myACHReturns.status + ", " + myACHReturns.details);


    //list ACHReturns if there are any
    if (myACHReturns.totalNumRecords>0)
    {
      DefaultListModel myList = new DefaultListModel();

      jList1.setVisibleRowCount(myACHReturns.totalNumRecords);
      for (int i=0; i<myACHReturns.totalNumRecords; i++)
        myList.addElement("FrontEndTrace:" +
            myACHReturns.achReturnRecords.getACHReturnRecord().get(i).frontEndTrace +
                     ", EffectiveDate:" +
            myACHReturns.achReturnRecords.getACHReturnRecord().get(i).effectiveDate.toString() +
                     ", Name:" +
           myACHReturns.achReturnRecords.getACHReturnRecord().get(i).customerName +
                           ", Amount:" +
            myACHReturns.achReturnRecords.getACHReturnRecord().get(i).transAmount.toString() +
                           ", ResponseCode:" +
            myACHReturns.achReturnRecords.getACHReturnRecord().get(i).responseCode +
                           ", ActionDetail:" +
            myACHReturns.achReturnRecords.getACHReturnRecord().get(i).actionDetail +
                           ", ActionDate:" +
            myACHReturns.achReturnRecords.getACHReturnRecord().get(i).actionDate.toString());


      jList1.setModel(myList);
     }
  }
  catch (Exception e)
  {
     jTextField1.setText("Exception caught: " + e);
  }
    }
```

getresultfile

```
private void jButton6MouseClicked(java.awt.event.MouseEvent evt) {
  //need to 'import' the following:
  //import java.util.*;
  //import java.text.*;
  //import javax.xml.datatype.DatatypeConfigurationException;
  //import javax.xml.datatype.DatatypeFactory;
  //import javax.xml.datatype.XMLGregorianCalendar;
  //import javax.swing.DefaultListModel;

  try
  {
    //instances
    ACHWorksWS myACHWS = new ACHWorksWS();
    CompanyInfo myCompanyInfo = new CompanyInfo();
    ResultFile myResultFile = new ResultFile();
    DatatypeFactory myDTF;


    //assign values for myCompanyInfo
    myCompanyInfo.sss="TST";
    myCompanyInfo.locID="9505";
    myCompanyInfo.company="THAT COMPANY";
    myCompanyInfo.companyKey="RICO";
```

```
      try
      {
         //instance
         myDTF = DatatypeFactory.newInstance();

         //call GetResultFile method
         myResultFile =
              myACHWS.getACHWorksWSSoap().getResultFile(myCompanyInfo,myDTF.newXMLGregorianCalendar("2010-
              03-25T00:00:00"),myDTF.newXMLGregorianCalendar("2010-03-26T00:00:00"));
      }
      catch (DatatypeConfigurationException e)
      {
         jTextField1.setText("Date exception caught: " + e);
      }

      //assign portion of result to jTextField
      jTextField1.setText(myResultFile.status + ", " + myResultFile.details);


      //list past connection results if there are any
      if (myResultFile.totalResultRecords>0)
      {
        DefaultListModel myList = new DefaultListModel();

        jList1.setVisibleRowCount(myResultFile.totalResultRecords);
        for (int i=0; i<myResultFile.totalResultRecords; i++)
        {
          myList.addElement("DateTime:" +
              myResultFile.transResults.getTransResult().get(i).callDateTime.toString() +
                             ", Method" + myResultFile.transResults.getTransResult().get(i).callMethod +
                             ", Status" + myResultFile.transResults.getTransResult().get(i).status +
                             ", FileName:" + myResultFile.transResults.getTransResult().get(i).fileName);
          for (int j=0; j<myResultFile.transResults.getTransResult().get(i).totalNumErrors; j++)
             myList.addElement(">" +
              myResultFile.transResults.getTransResult().get(i).errors.getString().get(j));
        }

        jList1.setModel(myList);
       }
    }
    catch (Exception e)
    {
       jTextField1.setText("Exception caught: " + e);
    }
}
```

geterrorfile

```
private void jButton7MouseClicked(java.awt.event.MouseEvent evt) {
   //need to 'import' the following:
   //import java.util.*;
   //import java.text.*;
   //import javax.xml.datatype.DatatypeConfigurationException;
   //import javax.xml.datatype.DatatypeFactory;
```

```
//import javax.xml.datatype.XMLGregorianCalendar;
//import javax.swing.DefaultListModel;

try
{
  //instances
  ACHWorksWS myACHWS = new ACHWorksWS();
  CompanyInfo myCompanyInfo = new CompanyInfo();
  ErrorFile myErrorFile = new ErrorFile();
  DatatypeFactory myDTF;


  //assign values for myCompanyInfo
  myCompanyInfo.sss="TST";
  myCompanyInfo.locID="9505";
  myCompanyInfo.company="THAT COMPANY";
  myCompanyInfo.companyKey="RICO";

  try
  {
     //instance
     myDTF = DatatypeFactory.newInstance();

     //call GetResultFile method
     myErrorFile =
         myACHWS.getACHWorksWSSoap().getErrorFile(myCompanyInfo,myDTF.newXMLGregorianCalendar("2010-
         03-25T00:00:00"),myDTF.newXMLGregorianCalendar("2010-03-26T00:00:00"));
  }
  catch (DatatypeConfigurationException e)
  {
     jTextField1.setText("Date exception caught: " + e);
  }

  //assign portion of result to jTextField
  jTextField1.setText(myErrorFile.status + ", " + myErrorFile.details);


  //list past connection results if there are any
  if (myErrorFile.totalErrorRecords>0)
  {
    DefaultListModel myList = new DefaultListModel();

    jList1.setVisibleRowCount(myErrorFile.totalErrorRecords);
    for (int i=0; i<myErrorFile.totalErrorRecords; i++)
    {
      myList.addElement("DateTime:" +
                  myErrorFile.errorRecords.getErrorRecord().get(i).callDateTime.toString() +
                        ", Method" + myErrorFile.errorRecords.getErrorRecord().get(i).callMethod +
                        ", FileName:" + myErrorFile.errorRecords.getErrorRecord().get(i).fileName +
                        ", No Of Errors:" +
                           myErrorFile.errorRecords.getErrorRecord().get(i).totalNumErrors);
      for (int j=0; j<myErrorFile.getErrorRecords().getErrorRecord().get(i).totalNumErrors; j++)
         myList.addElement(">" +
         myErrorFile.errorRecords.getErrorRecord().get(i).errors.getString().get(j));
    }
```

```
         jList1.setModel(myList);
        }
    }
    catch (Exception e)
    {
       jTextField1.setText("Exception caught: " + e);
    }
}
```