Wolters Kluwer
CCH

CCH® SureTax

Communications, Energy and General Merchandise
Tax Rating Solution

Web Request
Application Programming Interface (API)

Revision: 2.2.2

8-17-2015

CONFIDENTIAL NOTICE

Copyright 2015 CCH INCORPORATED. A Wolters Kluwer business. All Rights Reserved Worldwide. Material in this publication may not be reproduced or transmitted in any for or by any means without pior writtern permission.

TABLE OF CONTENTS

1 General Overview.......... 4
1.1 Overview of Request and Response Options.......... 4
2 Request and Response Specification – Communications or Utilities Transaction.......... 5
2.1 Request Layout.......... 5
2.1.1 JSON request format with sample data.......... 7
2.1.2 XML request format with sample data.......... 8
2.1.3 SOAP 1.1 request format with sample data.......... 9
2.1.4 SOAP 1.2 request format with sample data.......... 10
2.2 Response Messages – Taxes Summarized by Invoice.......... 12
2.2.1 Detailed Response Type.......... 12
2.2.1.1 JSON format with sample data.......... 13
2.2.1.2 XML response format with sample data.......... 14
2.3 Response Messages – Taxes by Line Item.......... 17
2.3.1 Detailed Response Type.......... 17
2.3.1.1 JSON format with sample data.......... 18
2.3.1.2 XML format with sample data.......... 20
3 Request and Response Specification – General Sales, Communications or Utilities Transaction.......... 23
3.1 Request Layout.......... 23
3.1.1 JSON request format with sample data.......... 26
3.1.2 XML request format with sample data.......... 27
3.1.3 SOAP 1.1 request format with sample data.......... 28
3.1.4 SOAP 1.2 request format with sample data.......... 28
3.2 Response Messages – Taxes Summarized by Invoice.......... 29
3.2.1 Detailed Response Type.......... 29
3.2.1.1 JSON format with sample data.......... 30
3.2.1.2 XML response format with sample data.......... 31
3.3 Response Messages – Taxes by Line Item.......... 34
3.3.1 Detailed Response Type.......... 34
3.3.1.1 JSON format with sample data.......... 35
3.3.1.2 XML format with sample data.......... 36
4 Cancel Request Specification.......... 38
4.1 Cancel Request Layout.......... 38
4.1.1.1 HTTPS POST - JSON cancel request format with sample data.......... 38
4.1.1.2 HTTPS POST - XML cancel request format with sample data.......... 38
4.1.1.3 SOAP 1.1 cancel request format sample.......... 38
4.1.1.4 SOAP 1.2 cancel request format sample.......... 39
4.2 Cancel Request - Response Message.......... 40
4.2.1.1 JSON cancel - response format with sample data.......... 40
4.2.1.2 XML cancel - response format with sample data.......... 40
4.2.1.3 SOAP 1.1 cancel - response format with sample.......... 40
4.2.1.4 SOAP 1.2 cancel - response format with sample.......... 41
4.2.2 Implementation Examples.......... 41
4.2.2.1 PHP.......... 41
4.2.2.2 .NET.......... 42
5 Appendix A – Implementation Examples.......... 43
5.1 PHP.......... 43
5.1.1 Request using cURL.......... 43
5.1.2 SOAP request using cURL.......... 45
5.2 Microsoft .NET.......... 48
5.2.1 .NET Examples.......... 50
5.3 Using GZIP Compression.......... 51
6 Appendix B – Transaction Type Codes.......... 52
7 Appendix C – Provider Type (Regulatory) Codes.......... 53
8 Appendix D – Tax Exemption Record Indicators.......... 54
8.1 Specialty Exemptions.......... 56
9 Appendix E – Unit Type Codes.......... 57
10 Appendix F – Tax Type Codes.......... 58
11 Appendix G – Tax Authority Level Codes.......... 60

12 Appendix H – Response Code and Messages 61

12.1 Header Level Response Codes 61

12.2 Item Level Response Codes 61

13 Appendix I – Deprecated Features 63

13.1 Alternate Engine (V03) 63

14 Appendix J – Change History 64

1 General Overview

This Application Programming Interface (API) document defines the structures and methods that are used in the CCH SureTax tax rating and calculation system. The SureTax system can be used for tax calculation of services such as general sales, leasing, data, digital goods and telecommunications services (long distance, local, wireless, VOIP, private line, 800, 900, enhanced, etc.) and energy (utilities) services on a "real-time" basis by submitting web request/response messages via Internet services using HTTPS POST or SOAP.

For web requests submitted via HTTPS POST, the data should be formatted using either JSON or XML format. The response format type is the same as the request format.

For requests submitted via SOAP, responses are provided in same XML format as the request.

Please note that the current implementation is synchronous, where a response is provided for each request submitted.

Each request can contain one-to-many items (transaction records) which can correlate multiple customers and/or invoices.

Should you have any questions about this API documentation, please contact the CCH SureTax team at 1-877-480-4025 or e-mail the support team at support@suretax.com

1.1 Overview of Request and Response Options

The CCH SureTax API provides two distinct sets of calculation processes:

a. Communications or Utilities Transactions with responses summarized by invoice or by line item (see V01 & V02)

b. Consolidated General Merchandise / Communications / Utilities transactions with responses by invoice or line item (see V04 & V05)

2 Request and Response Specification – Communications or Energy Transaction

2.1 Request Layout

All POST field names are required and are case-sensitive. Field values are required unless otherwise noted.

Request Record

FIELD NAME	POST FIELD	DESCRIPTION
Client Number	ClientNumber	Client ID Number – provided by SureTax. Required. Max Len: 10
Business Unit	BusinessUnit	Client's Business Unit. Value for this field is not required. Max Len: 20
Validation Key	ValidationKey	Validation Key provided by SureTax. Required for client access to API function. Max Len: 36
Data Year	DataYear	Required. YYYY – Year to use for tax calculation purposes
Data Month	DataMonth	Required. MM – Month to use for tax calculation purposes. ***Leading zero is preferred.***
Total Revenue	TotalRevenue	Required. Format: $$$$$$$$$.CCCC *For Negative charges, the first position should have a minus '-' indicator.*
Return File Code	ReturnFileCode	Required. 0 – Default Q – Quote purposes – taxes are computed and returned in the response message for generating quotes. No detailed tax information is saved in the SureTax tables for reporting.
Client Tracking Field	ClientTracking	Field for client transaction tracking. This value will be provided in the response data. Value for this field is not required, but preferred. Max Len: 100
Industry Exemption	IndustryExemption	*Reserved for future use.*
Response Group	ResponseGroup	Required. Determines how taxes are grouped for the response. Values: 03 – Tax grouped by State + Customer Number + Invoice Number (Default) 13 – Tax grouped by State + Customer Number + Invoice Number (this response group is the same as "03" except the CCH Geocode is appended to the state in the StateCode field. For example: NY|US36001A0013)
Response Type	ResponseType	Required. Determines the granularity of taxes and (optionally) the decimal precision for the tax calculations and amounts in the response. First Position Values: D – Detailed. Tax values are returned by tax type for all levels of tax (Federal, State, and Local). (optional) Second Position Values: An integer (1-9) determines the decimal places used for all tax responses. If no value is provided, all taxes are returned with a default number decimal places. This default precision value varies by "engine". For example, to return tax amounts to four decimals places, the Response Type would be: D4
Client Items	ItemList	List of Item records

Item layout

FIELD NAME	POST FIELD	DESCRIPTION
Line Number	LineNumber	Used to identify an item within the request. If no value is provided, requests are numbered sequentially. Max Len: 40
Invoice Number	InvoiceNumber	Used for tax aggregation by Invoice. Must be alphanumeric. Max Len: 40
Customer Number	CustomerNumber	Used for tax aggregation by Customer. Must be alphanumeric. Max Len: 40
Origination Number	OrigNumber	Required when using Tax Situs Rule 01 or 03. Format: NPANXXNNNN
Termination Number	TermNumber	Required when using Tax Situs Rule 01. Format: NPANXXNNNN
Billed to Number	BillToNumber	Required when using Tax Situs Rule 01 or 02. Format: NPANXXNNNN
Billing Zip code	Zipcode	Required when using Tax Situs Rule 04, 05, or 14. Zip code in format: 99999 (US or US territory) or X9X9X9 (Canadian) For VAT (Tax Situs Rule 14) International country code in format: XX
Billing Zip code extension	Plus4	Zip code extension in format: 9999 (not applicable for Tax Situs Rule 14)
Secondary Zip code	P2PZipcode	Secondary zip code in format: 99999 (US or US territory) or X9X9X9 (Canadian)
Secondary Zip code extension	P2PPlus4	Secondary zip code extension in format: 99999 (US or US territory) or X9X9X9 (Canadian)
Date of Transaction	TransDate	Required. Date of transaction. Valid date formats include: MM/DD/YYYY MM-DD-YYYY YYYY-MM-DDTHH:MM:SS
Revenue	Revenue	Required. Format: $$$$$$$$$.CCCC *For Negative charges, the first position should have a minus '-'indicator.*
Units	Units	Required. Units representing number of "lines" or unique charges contained within the revenue. This value is essentially a multiplier on unit-based fees (e.g. E911 fees). Format: 99999. Default should be 1 (one unit).
Unit Type	UnitType	Required. **00** – Default / Number of unique access lines. *See **Appendix E** for additional values.
Call Duration	Seconds	Required. Duration of call in seconds. Format **99999**. Default should be **1.**
Tax Included Code	TaxIncludedCode	Required. Values: **0** – Default (No Tax Included) 1 – Tax Included in Revenue

FIELD NAME	POST FIELD	DESCRIPTION
Tax Situs Rule	TaxSitusRule	Required. Values: 01 – Two-out-of-Three test using NPA-NXX 02 – Billed to number 03 – Origination number 04 – Zip code 05 – Zip code + 4 07 – Point to Point Zip codes (private line transactions)* 09 – Two-out-of-three test using Zip+4 as tax situs jurisdiction 11 – Used when Zipcode/Plus4 field is assigned as the Billing location and P2PZipcode/P2PPlus4 assigned as Service location* 14 – Use Zip code field for international country code (VAT calculations) 17 - Point to Point Zip codes (private line transactions) with both A and Z endpoints calculated*
Transaction Type Code	TransTypeCode	Required. Transaction Type Indicator. **See Appendix A for values**.
Sales Type Code	SalesTypeCode	Required. Values: **R** – Residential customer (default) B – Business customer I – Industrial customer L – Lifeline customer
Regulatory Code	RegulatoryCode	Required. Provider Type. **See Appendix C for values**.
Tax Exemption Codes	TaxExemptionCodeList	Required. Tax Exemption to be applied to this item only. **See Appendix D for values**.

* **Notes**:

i. Per **Tax Situs rule of "07"** - Point to Point zip codes (Private Line transactions), unlike most telecommunications services, the tracking of which entails mapping charges therefor to a single sourcing location or service address, a sale of private-line service is unique in that, in many cases two destinations are involved. Thus, for *point-to-point* private-line service, **both** endpoints have to be provided separately in order to determine the collective tax liability of a given private line. As such, the default calculation of Private Line services (Line Charge, Connection/Disconnection and Service Charges) is a ***50% percent taxable*** factor on all taxes and surcharges for each point of the private line.

   Hence, the user's billing system must be programmed, first to calculate the tax liability for Point A of the private line, followed by running an identical tax calculation for Point Z of the same private line. This requires two sets of transactions to be provided to SureTax; one for Point A to Point Z, and a second for Point Z to Point A using the Zipcode and P2PZipcode fields. However, separate transaction type codes can be provided if a 100% percent taxable factor is desired for a specific Private Line service.

   Please see the "Private Line Overview" in the associated *Telecom Mapping Guide* for additional details.

ii. When the **Tax Situs rule of "11"** is selected, SureTax will use the P2PZipcode / P2PPlus4 for the specific SureTax Transaction Type codes designated to be taxed at the service location. Within the system, we have a client-specific table of Transaction Type codes that will trigger the system to use the P2PZipcode field for the jurisdiction assignment instead of the Zipcode field. This enables dynamic jurisdiction assignment by product/service to an alternate location (in this case the service location) for taxing. If a Transaction Type code is submitted that is not in the client-specific Service Location table, then the value within the Zipcode/Plus4 field is used.

iii. Unlike **Tax Situs rule "07"**, **Tax Situs rule "17"** will compute both endpoints of the private line with one input request. As such, one request may generate two response lines within different taxing jurisdictions.

2.1.1 JSON request format with sample data

For information on the JSON format - http://en.wikipedia.org/wiki/JSON

Note:
When supplying the JSON string as a POST variable, the request value would be supplied as "`request={…JSON…}`". The Headers Content-Type value would be provided as: **"Content-Type: application/x-www-form-urlencoded".**

When supplying JSON serialized, the request value would be supplied as: `{“request”:…JSON…”}`. The Headers Content-Type value would be provided as: “**Content-Type: application/json”.** This response will be returned wrapped in a “d” object. For more information on this approach, please see the following article: http://encosia.com/a-breaking-change-between-versions-of-aspnet-ajax/#comment-34045

```
{
“ClientNumber”:”000000000”,
“BusinessUnit”:”bizUnit”,
“ValidationKey”:”19491161-F004-4F44-BDB3-E976D6739A64”,
“DataYear”:”2014”,
“DataMonth”:”02”,
“TotalRevenue”:40.00,
“ReturnFileCode”:"0“,
”ClientTracking”:”track”,
”IndustryExemption”:”“,
”ResponseType”:”D”,
”ResponseGroup”:”03”,
”ItemList”:[
    {
    “LineNumber”:”1”,
    ”InvoiceNumber”:”1”,
    ”CustomerNumber”:”000000001”,
    ”OrigNumber”:”8585260000”,
    ”TermNumber”:”8585260000”,
    ”BillToNumber”:”8585260000”,
    ”Zipcode”:”“,
    ”Plus4”:”“,
    ”P2PZipcode”:”“,
    ”P2PPlus4”:”“,
    ”TransDate”:”2001-01-01”,
    ”Revenue”:40.00,
    ”Units”:1,
    ”UnitType”:”00“,
    ”Seconds”:55,
    ”TaxIncludedCode”:”0”,
    ”TaxSitusRule”:”01”,
    ”TransTypeCode”:”010101”,
    ”SalesTypeCode”:”R”,
    ”RegulatoryCode”:”99”,
    ”TaxExemptionCodeList”:[“00”,”00”]
    }]
}
```

2.1.2 XML request format with sample data

```
<?xml version=“1.0”?>
<Request xmlns:xsi=“http://www.w3.org/2001/XMLSchema-instance” xmlns:xsd=“http://www.w3.org/2001/XMLSchema”>
    <ClientNumber>000000000</ClientNumber>
    <BusinessUnit>bizUnit</BusinessUnit>
    <ValidationKey>19491161-F004-4F44-BDB3-E976D6739A64</ValidationKey>
    <DataYear>2014</DataYear>
    <DataMonth>02</DataMonth>
    <TotalRevenue>40.00</TotalRevenue>
    <ReturnFileCode>0</ReturnFileCode>
    <ClientTracking>track</ClientTracking>
    <IndustryExemption />
    <ResponseType>D</ResponseType>
    <ResponseGroup>03</ResponseGroup>
    <ItemList>
        <Item>
            <LineNumber>1</LineNumber>
            <InvoiceNumber>1</InvoiceNumber>
            <CustomerNumber>000000001</CustomerNumber>
            <OrigNumber>8585260000</OrigNumber>
            <TermNumber>8585260000</TermNumber>
            <BillToNumber>8585260000</BillToNumber>
            <Zipcode />
            <Plus4 />
            <P2PZipcode />
            <P2PPlus4 />
            <TransDate>2001-01-01T00:00:00</TransDate>
            <Revenue>40.00</Revenue>
            <Units>1</Units>
            <UnitType>00</UnitType>
            <Seconds>0</Seconds>
            <TaxIncludedCode>0</TaxIncludedCode>
            <TaxSitusRule>01</TaxSitusRule>
            <TransTypeCode>010101</TransTypeCode>
            <SalesTypeCode>R</SalesTypeCode>
            <RegulatoryCode>99</RegulatoryCode>
            <TaxExemptionCodeList>
                <string>00</string>
                <string>00</string>
            </TaxExemptionCodeList>
        </Item>
    </ItemList>
</Request>
```

2.1.3 SOAP 1.1 request format with sample data

```
POST Services/V01/SureTax.asmx HTTP/1.1
Host: testapi.taxrating.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/SoapRequest"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SoapRequest xmlns="http://tempuri.org/">
      <request>
        <ExtensionData />
        <BusinessUnit>string</BusinessUnit>
        <ClientNumber>string</ClientNumber>
        <ClientTracking>string</ClientTracking>
        <DataMonth>string</DataMonth>
        <DataYear>string</DataYear>
        <IndustryExemption>string</IndustryExemption>
        <ItemList>
          <Item>
            <ExtensionData xsi:nil="true" />
            <BillToNumber>string</BillToNumber>
            <CustomerNumber>string</CustomerNumber>
            <InvoiceNumber>string</InvoiceNumber>
            <LineNumber>string</LineNumber>
            <OrigNumber>string</OrigNumber>
            <P2PPlus4>string</P2PPlus4>
            <P2PZipcode>string</P2PZipcode>
            <Plus4>string</Plus4>
            <RegulatoryCode>string</RegulatoryCode>
            <Revenue>decimal</Revenue>
            <SalesTypeCode>string</SalesTypeCode>
            <Seconds>int</Seconds>
            <TaxExemptionCodeList xsi:nil="true" />
            <TaxIncludedCode>string</TaxIncludedCode>
            <TaxSitusOverrideCode>string</TaxSitusOverrideCode>
            <TaxSitusRule>string</TaxSitusRule>
            <TermNumber>string</TermNumber>
            <TransDate>string</TransDate>
            <TransTypeCode>string</TransTypeCode>
            <UnitType>string</UnitType>
            <Units>int</Units>
            <Zipcode>string</Zipcode>
          </Item>
          <Item>
            <ExtensionData xsi:nil="true" />
            <BillToNumber>string</BillToNumber>
            <CustomerNumber>string</CustomerNumber>
            <InvoiceNumber>string</InvoiceNumber>
            <LineNumber>string</LineNumber>
            <OrigNumber>string</OrigNumber>
            <P2PPlus4>string</P2PPlus4>
            <P2PZipcode>string</P2PZipcode>
            <Plus4>string</Plus4>
            <RegulatoryCode>string</RegulatoryCode>
            <Revenue>decimal</Revenue>
            <SalesTypeCode>string</SalesTypeCode>
            <Seconds>int</Seconds>
            <TaxExemptionCodeList xsi:nil="true" />
            <TaxIncludedCode>string</TaxIncludedCode>
            <TaxSitusOverrideCode>string</TaxSitusOverrideCode>
            <TaxSitusRule>string</TaxSitusRule>
            <TermNumber>string</TermNumber>
            <TransDate>string</TransDate>
            <TransTypeCode>string</TransTypeCode>
            <UnitType>string</UnitType>
            <Units>int</Units>
            <Zipcode>string</Zipcode>
          </Item>
        </ItemList>
        <ResponseGroup>string</ResponseGroup>
        <ResponseType>string</ResponseType>
        <ReturnFileCode>string</ReturnFileCode>
        <TotalRevenue>decimal</TotalRevenue>
        <ValidationKey>string</ValidationKey>
      </request>
    </SoapRequest>
  </soap:Body>
</soap:Envelope>
```

2.1.4 SOAP 1.2 request format with sample data

```
POST Services/V01/SureTax.asmx HTTP/1.1
Host: testapi.taxrating.net
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <SoapRequest xmlns="http://tempuri.org/">
      <request>
        <ExtensionData />
        <BusinessUnit>string</BusinessUnit>
        <ClientNumber>string</ClientNumber>
        <ClientTracking>string</ClientTracking>
        <DataMonth>string</DataMonth>
        <DataYear>string</DataYear>
        <IndustryExemption>string</IndustryExemption>
        <ItemList>
          <Item>
            <ExtensionData xsi:nil="true" />
            <BillToNumber>string</BillToNumber>
            <CustomerNumber>string</CustomerNumber>
            <InvoiceNumber>string</InvoiceNumber>
            <LineNumber>string</LineNumber>
            <OrigNumber>string</OrigNumber>
            <P2PPlus4>string</P2PPlus4>
            <P2PZipcode>string</P2PZipcode>
            <Plus4>string</Plus4>
            <RegulatoryCode>string</RegulatoryCode>
            <Revenue>decimal</Revenue>
            <SalesTypeCode>string</SalesTypeCode>
            <Seconds>int</Seconds>
            <TaxExemptionCodeList xsi:nil="true" />
            <TaxIncludedCode>string</TaxIncludedCode>
            <TaxSitusOverrideCode>string</TaxSitusOverrideCode>
            <TaxSitusRule>string</TaxSitusRule>
            <TermNumber>string</TermNumber>
            <TransDate>string</TransDate>
            <TransTypeCode>string</TransTypeCode>
            <UnitType>string</UnitType>
            <Units>int</Units>
            <Zipcode>string</Zipcode>
          </Item>
          <Item>
            <ExtensionData xsi:nil="true" />
            <BillToNumber>string</BillToNumber>
            <CustomerNumber>string</CustomerNumber>
            <InvoiceNumber>string</InvoiceNumber>
            <LineNumber>string</LineNumber>
            <OrigNumber>string</OrigNumber>
            <P2PPlus4>string</P2PPlus4>
            <P2PZipcode>string</P2PZipcode>
            <Plus4>string</Plus4>
            <RegulatoryCode>string</RegulatoryCode>
            <Revenue>decimal</Revenue>
            <SalesTypeCode>string</SalesTypeCode>
            <Seconds>int</Seconds>
            <TaxExemptionCodeList xsi:nil="true" />
            <TaxIncludedCode>string</TaxIncludedCode>
            <TaxSitusOverrideCode>string</TaxSitusOverrideCode>
            <TaxSitusRule>string</TaxSitusRule>
            <TermNumber>string</TermNumber>
            <TransDate>string</TransDate>
            <TransTypeCode>string</TransTypeCode>
            <UnitType>string</UnitType>
            <Units>int</Units>
            <Zipcode>string</Zipcode>
          </Item>
        </ItemList>
        <ResponseGroup>string</ResponseGroup>
        <ResponseType>string</ResponseType>
        <ReturnFileCode>string</ReturnFileCode>
        <TotalRevenue>decimal</TotalRevenue>
        <ValidationKey>string</ValidationKey>
      </request>
    </SoapRequest>
  </soap12:Body>
</soap12:Envelope>
```

2.2 Response Messages – Taxes Summarized by Invoice

For development and testing, please use the following URL:
https://testapi.taxrating.net/Services/V01/SureTax.asmx/PostRequest

The response message values are returned from the HTTPS POST API functions in JSON or XML format. The response is composed of tax response header and tax response items section.

Two types of response messages are available - summary or detailed. Both summary and detailed response types have the same tax response header. The difference between a summary or detailed response is the type of tax records returned within items of the response. For a summary response, the tax items will be summarized by Federal, State, and Local taxes.

For a detailed response, the tax items are not summarized - all tax amounts with tax types for the transaction will be provided. The list of valid tax type codes can be found in **Appendix F**. All tax amounts are provided at four decimal places.

Please note that the tax output will be grouped (state, invoice number, customer number) based on the ***ResponseGroup*** selection in the request.

The detailed response type is recommended given the granular tax information returned.

2.2.1 Detailed Response Type

The detailed response message contains the following elements:

Response Header

FIELD NAME	DESCRIPTION
Successful	Response will be either 'Y' or 'N' : **Y** = Success / Success with Item error **N** = Failure
ResponseCode	ResponseCode: **9999** – Request was successful. **1101-1400** – Range of values for a failed request (no processing occurred) **9001** – Request was successful, but items within the request have errors. The specific items with errors are provided in the *ItemMessages* field.
HeaderMessage	Response message: For *ResponseCode* **9999** – "Success" For *ResponseCode* **9001** – "Success with Item errors". See the *ItemMessages* field for a list of items / errors. For *ResponseCode* **1100-1400** – Unsuccessful / declined web request. See **Appendix H** for a list of the response code and messages.
ItemMessages	This field contains a list of items that were not able to be processed due to bad or invalid data (see Response Code of "9001"). These invalid items will be listed by line number with the corresponding response code and message. When an item error occurs, no tax processing will occur for that item record. When an error for an item occurs, this field contains the following fields: ▪ **LineNumber** – a value corresponding to the line number in the web request ▪ **ResponseCode** – a value in the range 9100-9400. ▪ **Message** – the error message corresponding to the ResponseCode See **Appendix H** for a list of the response code and messages. If no item errors exist, the field will have a value of **empty**.
ClientTracking	Client transaction tracking provided in web request.

TotalTax	Total Tax – a total of all taxes included in the *TaxList*
TransId	Transaction ID (integer) – provided by SureTax
GroupList	See Group (contains one-to-many Groups)

Group

FIELD NAME	DESCRIPTION
StateCode	Tax State
InvoiceNumber	Invoice Number
CustomerNumber	Customer number
TaxList	See Tax Item(s) – contains one-to-many Tax Items

Note: *The tax output will be grouped (state, invoice number, customer number) based on the* **ResponseGroup** *selection in the request. InvoiceNumber and/or CustomerNumber will only be returned if selected to be grouped.*

TaxItem(s)

FIELD NAME	DESCRIPTION
TaxTypeCode	Tax Type Code
TaxTypeDesc	Tax Type Description
TaxAmount	Tax Amount

2.2.1.1 JSON format with sample data

For JSON, the tag *StateCode* is always provided in the Group List. The values for *CustomerNumber* and/or *InvoiceNumber* are populated based on the *ResponseGroup* selection in the request. Otherwise, the values for *CustomerNumber* and *InvoiceNumber will* be returned as *null.*

Successful Example:

```
{
    "Successful":"Y",
    "ResponseCode":"9999",
    "HeaderMessage":"Success",
    "ItemMessages":[],
    "ClientTracking":"7310",
    "TotalTax":"26.53",
    "TransId":4366,
          "GroupList":
                      [{
                         "CustomerNumber":"00123",
                         "InvoiceNumber":"12345678",
                         "StateCode":"CA",
                         "TaxList":
                               [{
                                "TaxAmount":"10.45",
                                "TaxTypeCode":"316",
                                "TaxTypeDesc":"LOCAL UTILITY USERS TAX"},
                                {
                                "TaxAmount":"15.50",
                                "TaxTypeCode":"035",
                                "TaxTypeDesc":"FEDERAL UNIVERSAL SERVICE FUND"
                                },
                                {
                                "TaxAmount":"0.58",
                                "TaxTypeCode":"060",
                                "TaxTypeDesc":"FEDERAL TRS FUND"
                                }],
                       }],

}
</string>
```

Success with Item Errors Example:

```
{
    "Successful":"Y",
    "ResponseCode":"9001",
    "HeaderMessage":"Success with Item errors",
    "ItemMessages":[{ "LineNumber":"1",
                         "ResponseCode":"9220",
                         "Message":"Invalid Unit Type - Must be 00"
                      }],
    "ClientTracking":"7310",
    "TotalTax":"26.53",
    "TransId":4366,
          "GroupList":
                        [{
                           "CustomerNumber":"00123",
                           "InvoiceNumber":"12345678",
                           "StateCode":"CA",
                           "TaxList":
                                 [{
                                  "TaxAmount":"10.45",
                                  "TaxTypeCode":"316",
                                  "TaxTypeDesc":"LOCAL UTILITY USERS TAX"},
                                  {
                                  "TaxAmount":"15.50",
                                  "TaxTypeCode":"035",
                                  "TaxTypeDesc":"FEDERAL UNIVERSAL SERVICE FUND"
                                  },
                                  {
                                  "TaxAmount":"0.58",
                                  "TaxTypeCode":"060",
                                  "TaxTypeDesc":"FEDERAL TRS FUND"
                                  }],
                         }],

}
</string>
```

2.2.1.2 XML response format with sample data

For XML, the tag *StateCode* is always provided in the Group List. However, values for *CustomerNumber* and/or *InvoiceNumber* tags are only provided when the *ResponseGroup* selection in the request is set accordingly.

Note: The XML that is returned from SureTax is processed through the Microsoft.NET XMLSerializer. As such, the returning XML data is html encoded (http://www.w3.org/TR/2004/REC-xml-20040204/#syntax) and converted to a well-formed XML that is readily transported. On the client receiving end, the data should be de-serialized / decoded to reconstruct the XML object from the stream.

For example, the actual XML data submitted from SureTax will resemble:

```
<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://tempuri.org/">&lt;?xml version="1.0"?&gt;
&lt;Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"&gt;
&lt;ExtensionData /&gt;
&lt;Successful&gt;Y&lt;/Successful&gt;
&lt;ResponseCode&gt;9999&lt;/ResponseCode&gt;
&lt;HeaderMessage&gt;Success&lt;/HeaderMessage&gt;
&lt;ItemMessages /&gt;
…
```

Users of low-level languages should be aware of this methodology as it might require a parsing routine to convert the serialized codes. For example the code " **&lt** " should be converted to " **<** ".

2.2.1.2.1 Response Group = “03” Example (Group Taxes by State + Invoice Number + Customer Number)

```
<?xml version="1.0"?>
<Response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Successful>Y</Successful>
  <ResponseCode>9999</ResponseCode>
  <HeaderMessage>Success</HeaderMessage>
  <ClientTracking>track</ClientTracking>
  <TotalTax>0.94</TotalTax>
  <TransId>627</TransId>
  <GroupList>
    <Group>
      <StateCode>CA</StateCode>
      <InvoiceNumber>1</InvoiceNumber>
      <CustomerNumber>000000001</CustomerNumber>
      <TaxList>
        <Tax>
          <TaxTypeCode>106</TaxTypeCode>
          <TaxTypeDesc>CA EMERG TEL. USERS SURCHARGE</TaxTypeDesc>
          <TaxAmount>0.20</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>108</TaxTypeCode>
          <TaxTypeDesc>CA P.U.C. FEE</TaxTypeDesc>
          <TaxAmount>0.07</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>117</TaxTypeCode>
          <TaxTypeDesc>CA HIGH COST FUND(B) SURCHARGE</TaxTypeDesc>
          <TaxAmount>0.18</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>118</TaxTypeCode>
          <TaxTypeDesc>CA TELECONNECT FUND</TaxTypeDesc>
          <TaxAmount>0.03</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>122</TaxTypeCode>
          <TaxTypeDesc>CA UNIVERSAL LIFELINE SURCHARGE</TaxTypeDesc>
          <TaxAmount>0.46</TaxAmount>
        </Tax>
      </TaxList>
    </Group>
  </GroupList>
</Response>
```

2.2.1.2.2 Success with Item Errors Example

```
<?xml version="1.0"?>
<Response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Successful>Y</Successful>
  <ResponseCode>9001</ResponseCode>
  <HeaderMessage>Success</HeaderMessage>
  <ItemMessages>
        <ItemMessage>
                 <LineNumber>1</LineNumber>
                 <ResponseCode>9220</ResponseCode>
                 <Message>Invalid Unit Type - Must be 00</Message>
        </ItemMessage>
  </ItemMessages>
  <ClientTracking>track</ClientTracking>
  <TotalTax>0.94</TotalTax>
  <TransId>624</TransId>
  <GroupList>
    <Group>
      <StateCode>CA</StateCode>
      <TaxList>
        <Tax>
          <TaxTypeCode>106</TaxTypeCode>
          <TaxTypeDesc>CA EMERG TEL. USERS SURCHARGE</TaxTypeDesc>
          <TaxAmount>0.20</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>108</TaxTypeCode>
          <TaxTypeDesc>CA P.U.C. FEE</TaxTypeDesc>
          <TaxAmount>0.07</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>117</TaxTypeCode>
          <TaxTypeDesc>CA HIGH COST FUND(B) SURCHARGE</TaxTypeDesc>
          <TaxAmount>0.18</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>118</TaxTypeCode>
          <TaxTypeDesc>CA TELECONNECT FUND</TaxTypeDesc>
          <TaxAmount>0.03</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>122</TaxTypeCode>
          <TaxTypeDesc>CA UNIVERSAL LIFELINE SURCHARGE</TaxTypeDesc>
          <TaxAmount>0.46</TaxAmount>
        </Tax>
      </TaxList>
    </Group>
  </GroupList>
</Response>
```

2.2.1.2.3 Unsuccessful Transaction Example

```
<?xml version="1.0"?>
<Response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xmlns:xsd="http://www.w3.org/2001/XMLSchema">
         <Successful>N</Successful>
         <ResponseCode>1201</ResponseCode>
         <HeaderMessage> Failure - Invalid Response Type </HeaderMessage>
         <ClientTracking>test1</ClientTracking>
         <TransId>0</TransId>
</Response>
```

2.3 Response Messages – Taxes by Line Item

SureTax supports an alternate processing engine for users that require taxes to be returned by Line Item in addition to Customer Number and Invoice Number. This response message also includes additional detail values for each of the taxes applied to the line items.

To use this alternate engine, please use the following URL: **https://testapi.taxrating.net/Services/V02/SureTax.asmx**

For additional information on the layouts and configurations for development and testing purposes, see https://testapi.taxrating.net/Documentation/V02/SureTax.asmx (requires valid login credentials.)

Please note: When using the V02 engine, the Response Group value in the Request Header is ignored as the results are returned by **Line Item + State + Invoice Number + Customer Number.**

2.3.1 Detailed Response Type

The detailed response message contains the following elements:

Response Header

FIELD NAME	DESCRIPTION
Successful	Response will be either 'Y' or 'N' : **Y** = Success / Success with Item error **N** = Failure
ResponseCode	ResponseCode: **9999** – Request was successful. **1101-1400** – Range of values for a failed request (no processing occurred) **9001** – Request was successful, but items within the request have errors. The specific items with errors are provided in the *ItemMessages* field.
HeaderMessage	Response message: For *ResponseCode* **9999** – "Success" For *ResponseCode* **9001** – "Success with Item errors". See the *ItemMessages* field for a list of items / errors. For *ResponseCode* **1100-1400** – Unsuccessful / declined web request. See **Appendix H** for a list of the response code and messages.
ItemMessages	This field contains a list of items that were not able to be processed due to bad or invalid data (see Response Code of "9001"). These invalid items will be listed by line number with the corresponding response code and message. When an item error occurs, no tax processing will occur for that item record. When an error for an item occurs, this field contains the following fields: ▪ **LineNumber** – a value corresponding to the line number in the web request ▪ **ResponseCode** – a value in the range 9100-9400. ▪ **Message** – the error message corresponding to the ResponseCode See **Appendix H** for a list of the response codes and messages. If no item errors exist, the field will have a value of **empty**.
ClientTracking	Client transaction tracking provided in web request.
TotalTax	Total Tax – a total of all taxes included in the *TaxList*
TransId	Transaction ID (integer) – provided by SureTax
GroupList	See Group (contains one-to-many Groups)

Group

FIELD NAME	DESCRIPTION
LineNumber	Line Number from Request
StateCode	Tax State
InvoiceNumber	Invoice Number
CustomerNumber	Customer number
TaxList	See Tax Item(s) – contains one-to-many Tax Items

Note: *The tax output will be grouped (line number, state, invoice number, customer number.) The* ***ResponseGroup*** *selection in the request is ignored when using the V02 engine.*

TaxItem(s)

FIELD NAME	DESCRIPTION
TaxTypeCode	Tax Type Code
TaxTypeDesc	Jurisdiction-specific Tax Type Description
TaxAmount	Tax Amount (taxes returned with five decimal places)
Revenue	Source Revenue for Line Item
CountyName	County Name of taxing jurisdiction
CityName	City Name of taxing jurisdiction
TaxRate	Tax rate for tax type applied in decimal format (if zero, please see FeeRate)
PercentTaxable	Percentage of the tax that is taxable for the tax type in decimal format. This can be less than 100% in certain circumstances such as application of a Safe Harbor rate or a Private Line that is allocated to two or more points.
FeeRate	The unit based fee for the tax type in format $.CCCC (if zero, please see TaxRate)
RevenueBase	The effective revenue for the tax provided in the TaxAmount field. This amount can be different than the amount in the Revenue field when taxes are impacted by specific exemptions and/or tax on tax.
TaxOnTax	The amount of tax on tax attributed to the final amount of tax. Please note this amount is included in the TaxAmount field total and is provided here separately only for reference purposes.

2.3.1.1 JSON format with sample data

```
{
    "Successful": "Y",
    "ResponseCode": "9999",
    "HeaderMessage": "Success",
    "ItemMessages": [],
    "ClientTracking": "Test Telecom",
    "TotalTax": "25.864212",
    "TransId": 2477023,
    "GroupList": [
        {
            "LineNumber": "1",
            "StateCode": "TX",
            "InvoiceNumber": "101",
            "CustomerNumber": "90120302",
            "TaxList": [
                {
                    "TaxTypeCode": "101",
                    "TaxTypeDesc": "STATE SALES TAX",
                    "TaxAmount": "7.220209",
                    "Revenue": "100",
                    "CountyName": "COLLIN",
                    "CityName": "PLANO",
                    "TaxRate": 0.0625,
                    "PercentTaxable": 1,
                    "FeeRate": 0,
                    "RevenueBase": "115.523344",
                    "TaxOnTax": "0.970209"
                },
                {
                    "TaxTypeCode": "403",
                    "TaxTypeDesc": "DALLAS MTA",
                    "TaxAmount": "1.155233",
                    "Revenue": "100",
                    "CountyName": "COLLIN",
                    "CityName": "PLANO",
                    "TaxRate": 0.01,
                    "PercentTaxable": 1,
                    "FeeRate": 0,
                    "RevenueBase": "115.523300",
                    "TaxOnTax": "0.155233"
                },
                {
                    "TaxTypeCode": "304",
                    "TaxTypeDesc": "CITY SALES TAX",
                    "TaxAmount": "1.155233",
                    "Revenue": "100",
                    "CountyName": "COLLIN",
                    "CityName": "PLANO",
                    "TaxRate": 0.01,
                    "PercentTaxable": 1,
                    "FeeRate": 0,
                    "RevenueBase": "115.523300",
                    "TaxOnTax": "0.155233"
                },
                {
                    "TaxTypeCode": "106",
                    "TaxTypeDesc": "9-1-1 EQUALIZATION FEE",
                    "TaxAmount": "0.060000",
                    "Revenue": "100",
                    "CountyName": "COLLIN",
                    "CityName": "PLANO",
                    "TaxRate": 0,
                    "PercentTaxable": 1,
                    "FeeRate": 0.06,
                    "RevenueBase": "100",
                    "TaxOnTax": "0"
                },
                {
                    "TaxTypeCode": "324",
                    "TaxTypeDesc": "MUNICIPAL RIGHT-OF-WAY FEE",
                    "TaxAmount": "4.230000",
                    "Revenue": "100",
                    "CountyName": "COLLIN",
                    "CityName": "PLANO",
                    "TaxRate": 0,
                    "PercentTaxable": 1,
                    "FeeRate": 4.23,
                    "RevenueBase": "100",
                    "TaxOnTax": "0"
                },
                {
                    "TaxTypeCode": "333",
                    "TaxTypeDesc": "PLANO 9-1-1 FEE",
                    "TaxAmount": "0.750000",
                    "Revenue": "100",
                    "CountyName": "COLLIN",
                    "CityName": "PLANO",
                    "TaxRate": 0,
                    "PercentTaxable": 1,
                    "FeeRate": 0.75,
                    "RevenueBase": "100",
                    "TaxOnTax": "0"
                },
                {
                    "TaxTypeCode": "035",
                    "TaxTypeDesc": "FEDERAL UNIVERSAL SERVICE FEE",
                    "TaxAmount": "10.231877",
                    "Revenue": "100",
                    "CountyName": "COLLIN",
                    "CityName": "PLANO",
                    "TaxRate": 0.156,
                    "PercentTaxable": 0.649,
                    "FeeRate": 0,
                    "RevenueBase": "65.588955",
                    "TaxOnTax": "0.107477"
                },
                {
                    "TaxTypeCode": "060",
                    "TaxTypeDesc": "Federal Regulatory Assessment Fee",
                    "TaxAmount": "1.061660",
                    "Revenue": "100",
                    "CountyName": "COLLIN",
                    "CityName": "PLANO",
                    "TaxRate": 0.01484,
                    "PercentTaxable": 0.649,
                    "FeeRate": 0,
                    "RevenueBase": "71.540431",
                    "TaxOnTax": "0.098544"
                }
            ]
        }
    ]
}
```

2.3.1.2 XML format with sample data

Note: The XML that is returned from SureTax is processed through the Microsoft.NET XMLSerializer. As such, the returning XML data is html encoded (http://www.w3.org/TR/2004/REC-xml-20040204/#syntax) and converted to a well-formed XML that is readily transported. On the client receiving end, the data should be deserialized / decoded to reconstruct the XML object from the stream.

For example, the actual XML data submitted from SureTax will resemble:

```
<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://tempuri.org/">&lt;?xml version="1.0"?&gt;
&lt;Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"&gt;
&lt;ExtensionData /&gt;
&lt;Successful&gt;Y&lt;/Successful&gt;
&lt;ResponseCode&gt;9999&lt;/ResponseCode&gt;
&lt;HeaderMessage&gt;Success&lt;/HeaderMessage&gt;
&lt;ItemMessages /&gt;
…
```

Users of low-level languages should be aware of this methodology as it might require a parsing routine to convert the serialized codes. For example the code " **&lt** " should be converted to " **<** ".

```
<?xml version="1.0"?>
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <ExtensionData />
  <Successful>Y</Successful>
  <ResponseCode>9999</ResponseCode>
  <HeaderMessage>Success</HeaderMessage>
  <ItemMessages />
  <ClientTracking>Test Telecom</ClientTracking>
  <TotalTax>25.864212</TotalTax>
  <TransId>2477622</TransId>
  <GroupList>
    <Group>
      <ExtensionData />
      <LineNumber>1</LineNumber>
      <StateCode>TX|US48085A0003</StateCode>
      <InvoiceNumber>101</InvoiceNumber>
      <CustomerNumber>80473921</CustomerNumber>
      <TaxList>
        <Tax>
          <ExtensionData />
          <TaxTypeCode>101</TaxTypeCode>
          <TaxTypeDesc>STATE SALES TAX</TaxTypeDesc>
          <TaxAmount>7.220209</TaxAmount>
          <Revenue>100</Revenue>
          <CountyName>COLLIN</CountyName>
          <CityName>PLANO</CityName>
          <TaxRate>0.062500000000</TaxRate>
          <PercentTaxable>1.000000</PercentTaxable>
          <FeeRate>0</FeeRate>
          <RevenueBase>115.523344</RevenueBase>
          <TaxOnTax>0.970209</TaxOnTax>
        </Tax>
        <Tax>
          <ExtensionData />
          <TaxTypeCode>403</TaxTypeCode>
          <TaxTypeDesc>DALLAS MTA</TaxTypeDesc>
          <TaxAmount>1.155233</TaxAmount>
          <Revenue>100</Revenue>
          <CountyName>COLLIN</CountyName>
          <CityName>PLANO</CityName>
          <TaxRate>0.010000000000</TaxRate>
          <PercentTaxable>1.000000</PercentTaxable>
          <FeeRate>0</FeeRate>
          <RevenueBase>115.523300</RevenueBase>
          <TaxOnTax>0.155233</TaxOnTax>
        </Tax>
        <Tax>
          <ExtensionData />
          <TaxTypeCode>304</TaxTypeCode>
          <TaxTypeDesc>CITY SALES TAX</TaxTypeDesc>
          <TaxAmount>1.155233</TaxAmount>
          <Revenue>100</Revenue>
          <CountyName>COLLIN</CountyName>
          <CityName>PLANO</CityName>
          <TaxRate>0.010000000000</TaxRate>
          <PercentTaxable>1.000000</PercentTaxable>
          <FeeRate>0</FeeRate>
          <RevenueBase>115.523300</RevenueBase>
          <TaxOnTax>0.155233</TaxOnTax>
        </Tax>
        <Tax>
          <ExtensionData />
          <TaxTypeCode>106</TaxTypeCode>
          <TaxTypeDesc>9-1-1 EQUALIZATION FEE</TaxTypeDesc>
          <TaxAmount>0.060000</TaxAmount>
          <Revenue>100</Revenue>
          <CountyName>COLLIN</CountyName>
          <CityName>PLANO</CityName>
          <TaxRate>0</TaxRate>
          <PercentTaxable>1.000000</PercentTaxable>
          <FeeRate>0.06000000</FeeRate>
          <RevenueBase>100</RevenueBase>
          <TaxOnTax>0</TaxOnTax>
        </Tax>
        <Tax>
          <ExtensionData />
          <TaxTypeCode>324</TaxTypeCode>
          <TaxTypeDesc>MUNICIPAL RIGHT-OF-WAY FEE</TaxTypeDesc>
          <TaxAmount>4.230000</TaxAmount>
          <Revenue>100</Revenue>
          <CountyName>COLLIN</CountyName>
          <CityName>PLANO</CityName>
          <TaxRate>0</TaxRate>
          <PercentTaxable>1.000000</PercentTaxable>
          <FeeRate>4.23000000</FeeRate>
          <RevenueBase>100</RevenueBase>
          <TaxOnTax>0</TaxOnTax>
        </Tax>
        <Tax>
          <ExtensionData />
          <TaxTypeCode>333</TaxTypeCode>
          <TaxTypeDesc>PLANO 9-1-1 FEE</TaxTypeDesc>
          <TaxAmount>0.750000</TaxAmount>
          <Revenue>100</Revenue>
          <CountyName>COLLIN</CountyName>
          <CityName>PLANO</CityName>
          <TaxRate>0</TaxRate>
          <PercentTaxable>1.000000</PercentTaxable>
          <FeeRate>0.75000000</FeeRate>
          <RevenueBase>100</RevenueBase>
          <TaxOnTax>0</TaxOnTax>
        </Tax>
        <Tax>
          <ExtensionData />
          <TaxTypeCode>035</TaxTypeCode>
          <TaxTypeDesc>FEDERAL UNIVERSAL SERVICE FEE</TaxTypeDesc>
          <TaxAmount>10.231877</TaxAmount>
          <Revenue>100</Revenue>
          <CountyName>COLLIN</CountyName>
          <CityName>PLANO</CityName>
          <TaxRate>0.156000000000</TaxRate>
          <PercentTaxable>0.649000</PercentTaxable>
          <FeeRate>0</FeeRate>
          <RevenueBase>65.588955</RevenueBase>
          <TaxOnTax>0.107477</TaxOnTax>
        </Tax>
        <Tax>
          <ExtensionData />
          <TaxTypeCode>060</TaxTypeCode>
          <TaxTypeDesc>Federal Regulatory Assessment Fee</TaxTypeDesc>
          <TaxAmount>1.061660</TaxAmount>
          <Revenue>100</Revenue>
          <CountyName>COLLIN</CountyName>
          <CityName>PLANO</CityName>
```

```
          <TaxRate>0.014840000000</TaxRate>
          <PercentTaxable>0.649000</PercentTaxable>
          <FeeRate>0</FeeRate>
          <RevenueBase>71.540431</RevenueBase>
          <TaxOnTax>0.098544</TaxOnTax>
        </Tax>
      </TaxList>
    </Group>
  </GroupList>
</Response>
```

3 Request and Response Specification – General Sales, Communications or Energy Transaction

3.1 Request Layout

All POST field names are required and are case-sensitive. Field values are required unless otherwise noted.

Request Record

FIELD NAME	POST FIELD	DESCRIPTION
Client Number	ClientNumber	Client ID Number – provided by SureTax. Required. Max Len: 10
Business Unit	BusinessUnit	Client's Business Unit. Value for this field is not required. Max Len: 20
Validation Key	ValidationKey	Validation Key provided by SureTax. Required for client access to API function. Max Len: 36
Data Year	DataYear	Required. YYYY – Year to use for tax calculation purposes
Data Month	DataMonth	Required. MM – Month to use for tax calculation purposes. ***Leading zero is preferred.***
Total Revenue	TotalRevenue	Format: $$$$$$$$$.CCCC *For Negative charges, the first position should have a minus '-' indicator.*
Return File Code	ReturnFileCode	Required. 0 – Default Q – Quote purposes – taxes are computed and returned in the response message for generating quotes. No detailed tax information is saved in the SureTax tables for reporting.
Client Tracking Field	ClientTracking	Field for client transaction tracking. This value will be provided in the response data. Value for this field is not required, but preferred. Max Len: 100
Industry Exemption	IndustryExemption	*Reserved for future use.*
Response Group	ResponseGroup	Required. Determines how taxes are grouped for the response. Values: 03 – Tax grouped by State + Customer Number + Invoice Number (Default) 13 – Tax grouped by State + Customer Number + Invoice Number (this response group is the same as "03" except the CCH Geocode is appended to the state in the StateCode field. For example: NY|US36001A0013)
Response Type	ResponseType	Required. Determines the granularity of taxes and (optionally) the decimal precision for the tax amounts in the response. First Position Values: D – Detailed. Tax values are returned by tax type for all levels of tax (Federal, State, and Local). (optional) Second Position Values: An integer (1-9) determines the decimal places used for all tax responses. If no value is provided, all taxes are returned to two decimal places. For example, to return tax amounts to four decimals places, the Response Type would be: D4

FIELD NAME	POST FIELD	DESCRIPTION
System Trace Audit Number	STAN	Optional. A unique value provided by client for transaction audit purposes. Max Len: 16
Client Items	ItemList	List of Item records

Item layout

FIELD NAME	POST FIELD	DESCRIPTION
Line Number	LineNumber	Used to identify an item within the request. If no value is provided, requests are numbered sequentially. Max Len: 40
Invoice Number	InvoiceNumber	Used for tax aggregation by Invoice. Must be alphanumeric. Max Len: 40
Customer Number	CustomerNumber	Used for tax aggregation by Customer. Must be alphanumeric. Max Len: 40
Origination Number	OrigNumber	Required when using Tax Situs Rule 01. Format: NPANXXNNNN
Termination Number	TermNumber	Required when using Tax Situs Rule 01. Format: NPANXXNNNN
Billed to Number	BillToNumber	Required when using Tax Situs Rule 01 or 02. Format: NPANXXNNNN
Billing / Ship to zip code	Zipcode	Required when using Tax Situs Rule 05. Zip code format: 99999 (US or US territory) or X9X9X9 (Canadian)
Billing Zip / Ship to zip code extension	Plus4	Zip code extension in format: 9999 If Plus4 value is not used or available, value should be “0000”.
Secondary zip code	P2PZipcode	Secondary Zip code in format: 99999 (US) or X9X9X9 (Canadian) If Plus4 value is not used or available, value should be “0000”.
Secondary zip code extension	P2PPlus4	Secondary zip code extension in format: 99999 (US) or X9X9X9 (Canadian)
Ship from zip code	ShipFromZipcode	Ship from code in format: 99999 (US) or X9X9X9 (Canadian) If Plus4 value is not used or available, value should be “0000”.
Ship from zip code extension	ShipFromPlus4	Ship from zip code extension in format: 99999 (US) or X9X9X9 (Canadian)
Order Placement zip code	OrderPlacementZipcode	Order Placement Zip code in format: 99999 (US) or X9X9X9 (Canadian) If this value is not provided, Billing / Ship to zip code will be assumed.
Order Placement Zip code extension	OrderPlacementPlus4	If Plus4 value is not used or available, value should be “0000”.
Order Approval Zip code	OrderApprovalZipcode	Order Approval Zip code in format: 99999 (US) or X9X9X9 (Canadian) If this value is not provided, Billing / Ship to zip code will be assumed.
Order Approval Zip code extension	OrderApprovalPlus4	If Plus4 value is not used or available, value should be “0000”.
CCH Geocode	Geocode	Required when using Tax Situs Rule 13. Allows the CCH Geocode to be submitted for jurisdiction assignment.
Date of Transaction	TransDate	Required. Date of transaction. Valid date formats include: MM/DD/YYYY MM-DD-YYYY
Revenue	Revenue	Required. Format: $$$$$$$$$.CCCC *For Negative charges, the first position should have a minus ‘-‘indicator.*

FIELD NAME	POST FIELD	DESCRIPTION
Units	Units	Required. Units representing number of "lines" or unique charges contained within the revenue. This value is essentially a multiplier on unit-based fees (e.g. E911 fees). Format: 99999. Default should be 1 (one unit).
Unit Type	UnitType	Required. 00 – Default / Number of unique access lines. *See Appendix E for additional values.
Call Duration	Seconds	Required. Duration of call in seconds. Format 99999. Default should be 1.
Tax Included Code	TaxIncludedCode	Required. Values: 0 – Default (No Tax Included) 1 – Tax Included in Revenue
Tax Situs Rule	TaxSitusRule	Required. Values: 01 – Two-out-of-Three test using NPA-NXX 02 – Billed to number 04 – Zip code 05 – Zip code + 4 07 – Point to Point Zip codes (private line transactions)* 09 – Two-out-of-three test using Zip+4 as tax situs jurisdiction 11 – Used when Zipcode/Plus4 field is assigned as the Billing location and P2PZipcode/P2PPlus4 assigned as Service location* 13 – CCH Geocode 15 – VAT using Destination Country Code 17 - Point to Point Zip codes (private line transactions) with both A and Z endpoints calculated*
Transaction Type Code	TransTypeCode	Required. Transaction Type Indicator. See Appendix A for values.
Sales Type Code	SalesTypeCode	Required. Values: R – Residential customer (default) B – Business customer I – Industrial customer L – Lifeline customer
Regulatory Code	RegulatoryCode	Required. Provider Type. See Appendix C for values.
Tax Exemption Codes	TaxExemptionCodeList	Required. Tax Exemption to be applied to this item only. See Appendix D for values.
Auxiliary Revenue	AuxRevenue	Use to supply "follow along" revenue items (such as freight) that adhere to the same taxability application as the source revenue for the transaction.
Auxiliary Revenue Type	AuxRevenueType	Values: 01 – Freight
Freight on Board	FreightOnBoard	Required. Values: D – Destination (default) O – Origination
Ship From Place of Business	ShipFromPOB	Required. Values: 1 – Yes (default) 0 – No
Mail Order	MailOrder	Required. Values: 1 – (default) *Reserved for future use.*
Common Carrier	CommonCarrier	Required. Values: 1 – Yes (default) 0 – No

FIELD NAME	POST FIELD	DESCRIPTION
Origination Country Code	OriginCountryCode	Required when using Tax Situs Rule 15. Origination International country code in format: XX
Destination Country Code	DestCountryCode	Required when using Tax Situs Rule 15. Destination International country code in format: XX This value will be use for the application of VAT.
User defined field	UDF	Field for client use at the item level. This value is not returned in the response, but is available for use in reports and extracts. Max Len: 100

* **Notes**:

i. Per **Tax Situs rule of "07"** - Point to Point zipcodes (Private Line transactions), unlike most telecommunications services, the tracking of which entails mapping charges therefor to a single sourcing location or service address, a sale of private-line service is unique in that, in many cases two destinations are involved. Thus, for *point-to-point* private-line service, **both** endpoints have to be provided separately in order to determine the collective tax liability of a given private line. As such, the default calculation of Private Line services (Line Charge, Connection/Disconnection and Service Charges) is a ***50% percent taxable*** factor on all taxes and surcharges for each point of the private line.

   Hence, the user's billing system must be programmed, first to calculate the tax liability for Point A of the private line, followed by running an identical tax calculation for Point Z of the same private line. This requires two sets of transactions to be provided to SureTax; one for Point A to Point Z, and a second for Point Z to Point A using the Zipcode and P2PZipcode fields. However, separate transaction type codes can be provided if a 100% percent taxable factor is desired for a specific Private Line service.

   Please see the "Private Line Overview" in the associated *Telecom Mapping Guide* for additional details.

ii. When the **Tax Situs rule of "11"** is selected, SureTax will use the P2PZipcode / P2PPlus4 for the specific SureTax Transaction Type codes designated to be taxed at the service location. Within the system, we have a client-specific table of Transaction Type codes that will trigger the system to use the P2PZipcode field for the jurisdiction assignment instead of the Zipcode field. This enables dynamic jurisdiction assignment by product/service to an alternate location (in this case the service location) for taxing. If a Transaction Type code is submitted that is not in the client-specific Service Location table, then the value within the Zipcode/Plus4 field is used.

iii. Unlike **Tax Situs rule "07", Tax Situs rule "17"** will compute both endpoints of the private line with one input request. As such, one request may generate two response lines within different taxing jurisdictions.

3.1.1 JSON request format with sample data

For information on the JSON format - http://en.wikipedia.org/wiki/JSON

Note:
When supplying the JSON string as a POST variable, the request value would be supplied as ‟request={…JSON…}”. The Headers Content-Type value would be provided as: "**Content-Type: application/x-www-form-urlencoded".**

When supplying JSON serialized, the request value would be supplied as: {‟request”:…JSON…”}. The Headers Content-Type value would be provided as: "**Content-Type: application/json".** This response will be returned wrapped in a "d" object. For more information on this approach, please see the following article:
http://encosia.com/a-breaking-change-between-versions-of-aspnet-ajax/#comment-34045

```
{
   "ClientNumber":"000000000",
   "BusinessUnit":"bizUnit ",
   "ValidationKey":"19491161-F004-4F44-BDB3-E976D6739A64",
   "DataYear":"2013",
   "DataMonth":"06",
   "TotalRevenue":100,
   "ReturnFileCode":"0",
   "ClientTracking":"Track",
   "IndustryExemption":"",
   "ResponseGroup":"13",
   "ResponseType":"D4",
   "STAN":1174114878,
   "ItemList":[
      {
         "LineNumber":"1",
         "InvoiceNumber":"302",
         "CustomerNumber":"2",
         "OrigNumber":"",
         "TermNumber":"",
         "BillToNumber":"",
         "Zipcode":"85364",
         "Plus4":"0000",
         "P2PZipcode":"",
         "P2PPlus4":"",
         "ShipFromZipcode":"85256",
         "ShipFromPlus4":"0000",
         "OrderPlacementZipcode":"85546",
         "OrderPlacementPlus4":"0000",
         "OrderApprovalZipcode":"85638",
         "OrderApprovalPlus4":"0000",
         "TransDate":"2013-06-30",
         "Revenue":100,
         "Units":1,
         "UnitType":"00",
         "Seconds":"1",
         "TaxIncludedCode":"0",
         "TaxSitusRule":"05",
         "TransTypeCode":"990101",
         "SalesTypeCode":"B",
         "RegulatoryCode":"99",
         "TaxExemptionCodeList":[“00”,”00”],
         "Geocode":"",
         "AuxRevenue":"0",
         "AuxRevenueType":"01",
         "FreightOnBoard":"D",
         "ShipFromPOB":"1",
         "MailOrder":"1",
         "CommonCarrier":"1",
         "OriginCountryCode":"US",
         "DestCountryCode":"US",
         "UDF":7521449127
      }
   ]
}
```

3.1.2 XML request format with sample data

```
<?xml version="1.0" encoding="UTF-8"?>
<Request xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <ClientNumber>000000000</ClientNumber>
   <BusinessUnit>bizUnit</BusinessUnit>
   <ValidationKey>19491161-F004-4F44-BDB3-E976D6739A64</ValidationKey>
   <DataYear>2013</DataYear>
   <DataMonth>06</DataMonth>
   <TotalRevenue>100</TotalRevenue>
   <ReturnFileCode>0</ReturnFileCode>
   <ClientTracking>Testing</ClientTracking>
   <IndustryExemption />
   <ResponseGroup>13</ResponseGroup>
   <ResponseType>D4</ResponseType>
   <STAN>3484788956</STAN>
   <ItemList>
      <Item>
         <LineNumber>1</LineNumber>
         <InvoiceNumber>302</InvoiceNumber>
         <CustomerNumber>2</CustomerNumber>
         <OrigNumber />
         <TermNumber />
         <BillToNumber />
         <Zipcode>85364</Zipcode>
         <Plus4>0000</Plus4>
         <P2PZipcode />
         <P2PPlus4 />
         <ShipFromZipcode>85256</ShipFromZipcode>
         <ShipFromPlus4>0000</ShipFromPlus4>
         <OrderPlacementZipcode>85546</OrderPlacementZipcode>
         <OrderPlacementPlus4>0000</OrderPlacementPlus4>
         <OrderApprovalZipcode>85638</OrderApprovalZipcode>
         <OrderApprovalPlus4>0000</OrderApprovalPlus4>
         <TransDate>2013-06-30</TransDate>
         <Revenue>100</Revenue>
         <Units>1</Units>
         <UnitType>00</UnitType>
         <Seconds>1</Seconds>
         <TaxIncludedCode>0</TaxIncludedCode>
         <TaxSitusRule>05</TaxSitusRule>
         <TransTypeCode>990101</TransTypeCode>
         <SalesTypeCode>B</SalesTypeCode>
         <RegulatoryCode>99</RegulatoryCode>
         <TaxExemptionCodeList>
            <string>00</string>
            <string>00</string>
         </TaxExemptionCodeList>
         <Geocode />
         <AuxRevenue>0</AuxRevenue>
         <AuxRevenueType>01</AuxRevenueType>
         <FreightOnBoard>D</FreightOnBoard>
         <ShipFromPOB>1</ShipFromPOB>
         <MailOrder>1</MailOrder>
         <CommonCarrier>1</CommonCarrier>
         <OriginCountryCode>US</OriginCountryCode>
         <DestCountryCode>US</DestCountryCode>
         <UDF>9416537848</UDF>
      </Item>
   </ItemList>
</Request>
```

3.1.3 SOAP 1.1 request format with sample data

To be added…

3.1.4 SOAP 1.2 request format with sample data

To be added…

3.2 Response Messages – Taxes Summarized by Invoice

For development and testing, please use the following URL:
https://testapi.taxrating.net/Services/V05/SureTax.asmx/PostRequest

The response message values are returned from the HTTPS POST API functions in JSON or XML format. The response is composed of tax response header and tax response items section.

Two types of response messages are available - summary or detailed. Both summary and detailed response types have the same tax response header. The difference between a summary or detailed response is the type of tax records returned within items of the response. For a summary response, the tax items will be summarized by Federal, State, and Local taxes.

For a detailed response, the tax items are not summarized - all tax amounts with tax types for the transaction will be provided. The list of valid tax type codes can be found in the Appendix. All tax amounts are provided at four decimal places.

Please note that the tax output will be grouped (state, invoice number, customer number) based on the ***ResponseGroup*** selection in the request.

The detailed response type is recommended given the granular tax information returned.

3.2.1 Detailed Response Type

The detailed response message contains the following elements:

Response Header

FIELD NAME	DESCRIPTION
Successful	Response will be either 'Y' or 'N' : **Y** = Success / Success with Item error **N** = Failure
ResponseCode	ResponseCode: **9999** – Request was successful. **1101-1400** – Range of values for a failed request (no processing occurred) **9001** – Request was successful, but items within the request have errors. The specific items with errors are provided in the *ItemMessages* field.
HeaderMessage	Response message: For *ResponseCode* **9999** – "Success" For *ResponseCode* **9001** – "Success with Item errors". See the *ItemMessages* field for a list of items / errors. For *ResponseCode* **1100-1400** – Unsuccessful / declined web request. See **Appendix H** for a list of the response code and messages.
ItemMessages	This field contains a list of items that were not able to be processed due to bad or invalid data (see Response Code of "9001"). These invalid items will be listed by line number with the corresponding response code and message. When an item error occurs, no tax processing will occur for that item record. When an error for an item occurs, this field contains the following fields: ▪ **LineNumber** – a value corresponding to the line number in the web request ▪ **ResponseCode** – a value in the range 9100-9400. ▪ **Message** – the error message corresponding to the ResponseCode See **Appendix H** for a list of the response code and messages. If no item errors exist, the field will have a value of **empty**.
ClientTracking	Client transaction tracking provided in web request.

TotalTax	Total Tax – a total of all taxes included in the *TaxList*
TransId	Transaction ID (integer) – provided by SureTax
STAN	Unique System Trace Audit Number (auto-assigned if one not provided)
GroupList	See Group (contains one-to-many Groups)

Group

FIELD NAME	DESCRIPTION
StateCode	Tax State
InvoiceNumber	Invoice Number
CustomerNumber	Customer number
TaxList	See Tax Item(s) – contains one-to-many Tax Items

Note: *The tax output will be grouped (state, invoice number, customer number) based on the* **ResponseGroup** *selection in the request. InvoiceNumber and/or CustomerNumber will only be returned if selected to be grouped.*

TaxItem(s)

FIELD NAME	DESCRIPTION
TaxTypeCode	Tax Type Code
TaxTypeDesc	Tax Type Description
TaxAmount	Tax Amount

3.2.1.1 JSON format with sample data

For JSON, the tag *StateCode* is always provided in the Group List. The values for *CustomerNumber* and/or *InvoiceNumber* are populated based on the *ResponseGroup* selection in the request. Otherwise, the values for *CustomerNumber* and *InvoiceNumber will* be returned as *null.*

Successful Example:

```
{
    "Successful":"Y",
    "ResponseCode":"9999",
    "HeaderMessage":"Success",
    "ItemMessages":[],
    "ClientTracking":"7310",
    "TotalTax":"26.53",
    "TransId":4366,
    "STAN":"6345218965",
          "GroupList":
                      [{
                         "CustomerNumber":"00123",
                         "InvoiceNumber":"12345678",
                         "StateCode":"CA",
                         "TaxList":
                               [{
                                "TaxAmount":"10.45",
                                "TaxTypeCode":"316",
                                "TaxTypeDesc":"LOCAL UTILITY USERS TAX"},
                                {
                                "TaxAmount":"15.50",
                                "TaxTypeCode":"035",
                                "TaxTypeDesc":"FEDERAL UNIVERSAL SERVICE FUND"
                                },
                                {
                                "TaxAmount":"0.58",
                                "TaxTypeCode":"060",
                                "TaxTypeDesc":"FEDERAL TRS FUND"
                                }],
                       }],

}
</string>
```

Success with Item Errors Example:

```
{
    "Successful":"Y",
    "ResponseCode":"9001",
    "HeaderMessage":"Success with Item errors",
    "ItemMessages":[{ "LineNumber":"1",
                        "ResponseCode":"9220",
                        "Message":"Invalid Unit Type - Must be 00"
                     }],
    "ClientTracking":"7310",
    "TotalTax":"26.53",
    "TransId":4366,
    "STAN":"6345218967",
          "GroupList":
                       [{
                          "CustomerNumber":"00123",
                          "InvoiceNumber":"12345678",
                          "StateCode":"CA",
                          "TaxList":
                                [{
                                 "TaxAmount":"10.45",
                                 "TaxTypeCode":"316",
                                 "TaxTypeDesc":"LOCAL UTILITY USERS TAX"},
                                 {
                                 "TaxAmount":"15.50",
                                 "TaxTypeCode":"035",
                                 "TaxTypeDesc":"FEDERAL UNIVERSAL SERVICE FUND"
                                 },
                                 {
                                 "TaxAmount":"0.58",
                                 "TaxTypeCode":"060",
                                 "TaxTypeDesc":"FEDERAL TRS FUND"
                                 }],
                        }],

}
</string>
```

3.2.1.2 XML response format with sample data

For XML, the tag *StateCode* is always provided in the Group List. However, values for *CustomerNumber* and/or *InvoiceNumber* tags are only provided when the *ResponseGroup* selection in the request is set accordingly.

Note: The XML that is returned from SureTax is processed through the Microsoft.NET XMLSerializer. As such, the returning XML data is html encoded (http://www.w3.org/TR/2004/REC-xml-20040204/#syntax) and converted to a well-formed XML that is readily transported. On the client receiving end, the data should be de-serialized / decoded to reconstruct the XML object from the stream.

For example, the actual XML data submitted from SureTax will resemble:

```
<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://tempuri.org/">&lt;?xml version="1.0"?&gt;
&lt;Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"&gt;
&lt;ExtensionData /&gt;
&lt;Successful&gt;Y&lt;/Successful&gt;
&lt;ResponseCode&gt;9999&lt;/ResponseCode&gt;
&lt;HeaderMessage&gt;Success&lt;/HeaderMessage&gt;
&lt;ItemMessages /&gt;
…
```

Users of low-level languages should be aware of this methodology as it might require a parsing routine to convert the serialized codes. For example the code " **&lt** " should be converted to " **<** ".

3.2.1.2.1 Response Group = “03” Example (Group Taxes by State + Invoice Number + Customer Number)

```xml
<?xml version="1.0"?>
<Response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Successful>Y</Successful>
  <ResponseCode>9999</ResponseCode>
  <HeaderMessage>Success</HeaderMessage>
  <ClientTracking>track</ClientTracking>
  <TotalTax>0.94</TotalTax>
  <TransId>627</TransId>
  <STAN>355023459</STAN>
  <GroupList>
    <Group>
      <StateCode>CA</StateCode>
      <InvoiceNumber>1</InvoiceNumber>
      <CustomerNumber>000000001</CustomerNumber>
      <TaxList>
        <Tax>
          <TaxTypeCode>106</TaxTypeCode>
          <TaxTypeDesc>CA EMERG TEL. USERS SURCHARGE</TaxTypeDesc>
          <TaxAmount>0.20</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>108</TaxTypeCode>
          <TaxTypeDesc>CA P.U.C. FEE</TaxTypeDesc>
          <TaxAmount>0.07</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>117</TaxTypeCode>
          <TaxTypeDesc>CA HIGH COST FUND(B) SURCHARGE</TaxTypeDesc>
          <TaxAmount>0.18</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>118</TaxTypeCode>
          <TaxTypeDesc>CA TELECONNECT FUND</TaxTypeDesc>
          <TaxAmount>0.03</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>122</TaxTypeCode>
          <TaxTypeDesc>CA UNIVERSAL LIFELINE SURCHARGE</TaxTypeDesc>
          <TaxAmount>0.46</TaxAmount>
        </Tax>
      </TaxList>
    </Group>
  </GroupList>
</Response>
```

3.2.1.2.2 Success with Item Errors Example

```xml
<?xml version="1.0"?>
<Response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Successful>Y</Successful>
  <ResponseCode>9001</ResponseCode>
  <HeaderMessage>Success</HeaderMessage>
  <ItemMessages>
         <ItemMessage>
                 <LineNumber>1</LineNumber>
                 <ResponseCode>9220</ResponseCode>
                 <Message>Invalid Unit Type - Must be 00</Message>
         </ItemMessage>
  </ItemMessages>
  <ClientTracking>track</ClientTracking>
  <TotalTax>0.94</TotalTax>
  <TransId>624</TransId>
  <STAN>355023469</STAN>
  <GroupList>
    <Group>
      <StateCode>CA</StateCode>
      <TaxList>
        <Tax>
          <TaxTypeCode>106</TaxTypeCode>
          <TaxTypeDesc>CA EMERG TEL. USERS SURCHARGE</TaxTypeDesc>
          <TaxAmount>0.20</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>108</TaxTypeCode>
          <TaxTypeDesc>CA P.U.C. FEE</TaxTypeDesc>
          <TaxAmount>0.07</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>117</TaxTypeCode>
          <TaxTypeDesc>CA HIGH COST FUND(B) SURCHARGE</TaxTypeDesc>
          <TaxAmount>0.18</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>118</TaxTypeCode>
          <TaxTypeDesc>CA TELECONNECT FUND</TaxTypeDesc>
          <TaxAmount>0.03</TaxAmount>
        </Tax>
        <Tax>
          <TaxTypeCode>122</TaxTypeCode>
          <TaxTypeDesc>CA UNIVERSAL LIFELINE SURCHARGE</TaxTypeDesc>
          <TaxAmount>0.46</TaxAmount>
        </Tax>
      </TaxList>
    </Group>
  </GroupList>
</Response>
```

3.2.1.2.3 Unsuccessful Transaction Example

```
<?xml version="1.0"?>
<Response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xmlns:xsd="http://www.w3.org/2001/XMLSchema">
        <Successful>N</Successful>
        <ResponseCode>1201</ResponseCode>
        <HeaderMessage> Failure - Invalid Response Type </HeaderMessage>
        <ClientTracking>test1</ClientTracking>
        <TransId>0</TransId>
        <STAN>355026459</STAN>

</Response>
```

3.3 Response Messages – Taxes by Line Item

SureTax supports an alternate processing engine for users that require taxes to be returned by Line Item in addition to Customer Number and Invoice Number. This response message also includes additional detail values for each of the taxes applied to the line items.

To use this alternate engine, please use the following URL: https://testapi.taxrating.net/Services/V04/SureTax.asmx

Please note: When using the V04 engine, the Response Group value in the Request Header is ignored as the results are returned by **Line Item + State + Invoice Number + Customer Number.**

3.3.1 Detailed Response Type

The detailed response message contains the following elements:

Response Header

FIELD NAME	DESCRIPTION
Successful	Response will be either 'Y' or 'N' : **Y** = Success / Success with Item error **N** = Failure
ResponseCode	ResponseCode: **9999** – Request was successful. **1101-1400** – Range of values for a failed request (no processing occurred) **9001** – Request was successful, but items within the request have errors. The specific items with errors are provided in the *ItemMessages* field.
HeaderMessage	Response message: For *ResponseCode* **9999** – "Success" For *ResponseCode* **9001** – "Success with Item errors". See the *ItemMessages* field for a list of items / errors. For *ResponseCode* **1100-1400** – Unsuccessful / declined web request. See **Appendix H** for a list of the response code and messages.
ItemMessages	This field contains a list of items that were not able to be processed due to bad or invalid data (see Response Code of "9001"). These invalid items will be listed by line number with the corresponding response code and message. When an item error occurs, no tax processing will occur for that item record. When an error for an item occurs, this field contains the following fields: ▪ **LineNumber** – a value corresponding to the line number in the web request ▪ **ResponseCode** – a value in the range 9100-9400. ▪ **Message** – the error message corresponding to the ResponseCode See **Appendix H** for a list of the response code and messages. If no item errors exist, the field will have a value of **empty**.
ClientTracking	Client transaction tracking provided in web request.
TotalTax	Total Tax – a total of all taxes included in the *TaxList*
TransId	Transaction ID (integer) – provided by SureTax
STAN	Unique System Trace Audit Number (auto-assigned if one not provided)
GroupList	See Group (contains one-to-many Groups)

Group

FIELD NAME	DESCRIPTION
LineNumber	Line Number from Request
StateCode	Tax State
InvoiceNumber	Invoice Number
CustomerNumber	Customer number
TaxList	See Tax Item(s) – contains one-to-many Tax Items

Note: *The tax output will be grouped (line number, state, invoice number, customer number.) The* ***ResponseGroup*** *selection in the request is ignored when using the V04 engine.*

TaxItem(s)

FIELD NAME	DESCRIPTION
TaxTypeCode	Tax Type Code
TaxTypeDesc	Jurisdiction-specific Tax Type Description
TaxAmount	Tax Amount (taxes returned with five decimal places)
Revenue	Source Revenue for Line Item
CountyName	County Name of taxing jurisdiction
CityName	City Name of taxing jurisdiction
TaxRate	Tax rate for tax type applied in decimal format (if zero, please see FeeRate)
PercentTaxable	Percentage of the tax that is taxable for the tax type in decimal format. This can be less than 100% in certain circumstances such as application of a Safe Harbor rate or a Private Line that is allocated to two or more points.
FeeRate	The unit based fee for the tax type in format $.CCCC (if zero, please see TaxRate)
RevenueBase	The effective revenue for the tax provided in the TaxAmount field. This amount can be different than the amount in the Revenue field when taxes are impacted by specific exemptions and/or tax on tax.
TaxOnTax	The amount of tax on tax attributed to the final amount of tax. Please note this amount is included in the TaxAmount field total and is provided here separately only for reference purposes.

3.3.1.1 JSON format with sample data

```
{
   "Successful":"Y",
   "ResponseCode":"9999",
   "HeaderMessage":"Success",
   "ItemMessages":[

   ],
   "ClientTracking":"Testing",
   "TotalTax":"8.4000",
   "TransId":375642,
   "STAN":"6345218965",
   "GroupList":[
      {
         "LineNumber":"1",
         "StateCode":"AZ|US0402785540",
         "InvoiceNumber":"302",
         "CustomerNumber":"2",
         "TaxList":[
            {
               "TaxTypeCode":"101",
               "TaxTypeDesc":"AZ TRANSACTION PRIVILEGE TAX",
               "TaxAmount":"5.6000",
               "Revenue":"100",
               "CountyName":"YUMA",
               "CityName":"YUMA PROVING GROUND",
               "TaxRate":0.056000000000,
               "PercentTaxable":1.000000,
               "FeeRate":0,
               "RevenueBase":"100.0000",
               "TaxOnTax":"0"
            },
            {
               "TaxTypeCode":"202",
               "TaxTypeDesc":"COUNTY TRANSACT. PRIVILEGE TAX",
               "TaxAmount":"1.1000",
               "Revenue":"100",
               "CountyName":"YUMA",
               "CityName":"YUMA PROVING GROUND",
               "TaxRate":0.011000000000,
               "PercentTaxable":1.000000,
               "FeeRate":0,
               "RevenueBase":"100.0000",
               "TaxOnTax":"0"
            },
            {
               "TaxTypeCode":"304",
               "TaxTypeDesc":"CITY TRANSACTION PRIVILEGE TAX",
               "TaxAmount":"1.7000",
               "Revenue":"100",
               "CountyName":"YUMA",
               "CityName":"YUMA PROVING GROUND",
               "TaxRate":0.017000000000,
               "PercentTaxable":1.000000,
               "FeeRate":0,
               "RevenueBase":"100.0000",
               "TaxOnTax":"0"
            }
         ]
      }
   ]
}
```

3.3.1.2 XML format with sample data

Note: The XML that is returned from SureTax is processed through the Microsoft.NET XMLSerializer. As such, the returning XML data is html encoded (http://www.w3.org/TR/2004/REC-xml-20040204/#syntax) and converted to a well-formed XML that is readily transported. On the client receiving end, the data should be deserialized / decoded to reconstruct the XML object from the stream.

For example, the actual XML data submitted from SureTax will resemble:

```
<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://tempuri.org/">&lt;?xml version="1.0"?&gt;
&lt;Response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"&gt;
&lt;ExtensionData /&gt;
&lt;Successful&gt;Y&lt;/Successful&gt;
&lt;ResponseCode&gt;9999&lt;/ResponseCode&gt;
&lt;HeaderMessage&gt;Success&lt;/HeaderMessage&gt;
&lt;ItemMessages /&gt;
…
```

Users of low-level languages should be aware of this methodology as it might require a parsing routine to convert the serialized codes. For example the code " **&lt** " should be converted to " **<** ".

```
<?xml version="1.0" encoding="UTF-8"?>
<Response xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <ExtensionData />
   <Successful>Y</Successful>
   <ResponseCode>9999</ResponseCode>
   <HeaderMessage>Success</HeaderMessage>
   <ItemMessages />
   <ClientTracking>Testing</ClientTracking>
   <TotalTax>8.4000</TotalTax>
   <TransId>375643</TransId>
   <STAN>355023458</STAN>
   <GroupList>
      <Group>
         <ExtensionData />
         <LineNumber>1</LineNumber>
         <StateCode>AZ|US0402785540</StateCode>
         <InvoiceNumber>302</InvoiceNumber>
         <CustomerNumber>2</CustomerNumber>
         <TaxList>
            <Tax>
               <ExtensionData />
               <TaxTypeCode>101</TaxTypeCode>
               <TaxTypeDesc>AZ TRANSACTION PRIVILEGE TAX</TaxTypeDesc>
               <TaxAmount>5.6000</TaxAmount>
               <Revenue>100</Revenue>
               <CountyName>YUMA</CountyName>
               <CityName>YUMA PROVING GROUND</CityName>
               <TaxRate>0.056000000000</TaxRate>
               <PercentTaxable>1.000000</PercentTaxable>
               <FeeRate>0</FeeRate>
               <RevenueBase>100.0000</RevenueBase>
               <TaxOnTax>0</TaxOnTax>
            </Tax>
            <Tax>
               <ExtensionData />
               <TaxTypeCode>202</TaxTypeCode>
               <TaxTypeDesc>COUNTY TRANSACT. PRIVILEGE TAX</TaxTypeDesc>
               <TaxAmount>1.1000</TaxAmount>
               <Revenue>100</Revenue>
               <CountyName>YUMA</CountyName>
               <CityName>YUMA PROVING GROUND</CityName>
               <TaxRate>0.011000000000</TaxRate>
               <PercentTaxable>1.000000</PercentTaxable>
               <FeeRate>0</FeeRate>
               <RevenueBase>100.0000</RevenueBase>
               <TaxOnTax>0</TaxOnTax>
            </Tax>
            <Tax>
               <ExtensionData />
               <TaxTypeCode>304</TaxTypeCode>
               <TaxTypeDesc>CITY TRANSACTION PRIVILEGE TAX</TaxTypeDesc>
               <TaxAmount>1.7000</TaxAmount>
               <Revenue>100</Revenue>
               <CountyName>YUMA</CountyName>
               <CityName>YUMA PROVING GROUND</CityName>
               <TaxRate>0.017000000000</TaxRate>
               <PercentTaxable>1.000000</PercentTaxable>
               <FeeRate>0</FeeRate>
               <RevenueBase>100.0000</RevenueBase>
               <TaxOnTax>0</TaxOnTax>
            </Tax>
         </TaxList>
      </Group>
   </GroupList>
</Response>
```

4 Cancel Request Specification

The Cancel Request method is used to cancel a previous web request tax calculation process. Only valid web requests for the current and prior calendar months can be cancelled. Transactions older than 60 days cannot be cancelled.

Caution: after a successful Cancel Request method is performed, all reporting data associated with the cancelled web request is removed from the system and will no longer be included on any reports.

For development and testing with HTTPS POST, please use the following URL:
https://testapi.taxrating.net/Services/V01/SureTax.asmx/CancelPostRequest

Note: This method exists with all "engines" (i.e. V01, V02, V04, and V05)

4.1 Cancel Request Layout

All fields are required unless otherwise noted. All field names are case-sensitive.

Request Record

FIELD NAME	POST FIELD	DESCRIPTION
Client Number	ClientNumber	Client ID Number – provided by SureTax.
Client Tracking Field	ClientTracking	Optional. Field for client transaction tracking.
Transaction ID	TransId	Transaction ID of web request to be cancelled.
Validation Key	ValidationKey	Validation Key provided by SureTax. Required for client access to API function.

4.1.1.1 HTTPS POST - JSON cancel request format with sample data

```
{
    "ClientNumber":"000000000",
    "ClientTracking":"test",
    "TransId":"999",
    "ValidationKey":"17491131-F004-4F44-BDB3-E979D6749A64"
}
```

4.1.1.2 HTTPS POST - XML cancel request format with sample data

```
<?xml version="1.0" encoding="utf-8"?>
<CancelRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
           <ClientNumber>000000000</ClientNumber>
           <ClientTracking>test</ClientTracking>
           <TransId>999</TransId>
           <ValidationKey>17491131-F004-4F44-BDB3-E979D6749A64</ValidationKey>
</CancelRequest>
```

4.1.1.3 SOAP 1.1 cancel request format sample

```
POST /V01/SureTax.asmx HTTP/1.1
Host: testapi.taxrating.net
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/CancelSoapRequest"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <CancelSoapRequest xmlns="http://tempuri.org/">
      <requestCancel>
        <ClientNumber>string</ClientNumber>
        <ClientTracking>string</ClientTracking>
        <TransId>int</TransId>
        <ValidationKey>string</ValidationKey>
      </requestCancel>
    </CancelSoapRequest>
  </soap:Body>
</soap:Envelope>
```

4.1.1.4 SOAP 1.2 cancel request format sample

```
POST /V01/SureTax.asmx HTTP/1.1
Host: testapi.taxrating.net
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <CancelSoapRequest xmlns="http://tempuri.org/">
      <requestCancel>
        <ClientNumber>string</ClientNumber>
        <ClientTracking>string</ClientTracking>
        <TransId>int</TransId>
        <ValidationKey>string</ValidationKey>
      </requestCancel>
    </CancelSoapRequest>
  </soap12:Body>
</soap12:Envelope>
```

4.2 Cancel Request - Response Message

Cancel Response Message

FIELD NAME	DESCRIPTION
Successful	Response will be either ‘Y' or ‘N' : **Y** = Success / Success with errors **N** = Failure
ResponseCode	ResponseCode: **9999** – Cancel Request was successful. **1101-1600** – Range of values for a failed request (no processing occurred). Values include: **1150** - Failure - Validation Key Required. **1151** - Failure - Invalid Validation Key. **1510** - Failure - Transaction is more than 60 days old. **9410** - Failure - Transaction is already cancelled.
HeaderMessage	Response message: For *ResponseCode* **9999** – “Success” For *ResponseCode* **9001** – “Success with Item errors”. See the *ItemMessages* field above for a list of items / errors. For *ResponseCode* **1100-1600** – Unsuccessful / declined web request. See above or (**Appendix H)** for a list.
ClientTracking	Field for client transaction tracking.
TransId	Transaction ID (integer) – provided by SureTax

4.2.1.1 JSON cancel - response format with sample data

```
<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://tempuri.org/">
          {
           "Successful":"Y",
           "ResponseCode":"9999",
           "HeaderMessage":"Success",
           "ClientTracking":"test",
           "TransId":0
          }
</string>
```

4.2.1.2 XML cancel - response format with sample data

```
<?xml version="1.0"?>
<CancelResponse xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <Successful>Y</Successful>
    <ResponseCode>9999</ResponseCode>
    <HeaderMessage>Success</HeaderMessage>
    <ClientTracking>test</ClientTracking>
    <TransId>0</TransId>
</CancelResponse>
```

4.2.1.3 SOAP 1.1 cancel - response format with sample

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <CancelSoapRequestResponse xmlns="http://tempuri.org/">
      <CancelSoapRequestResult>
```

```
        <Successful>string</Successful>
        <ResponseCode>string</ResponseCode>
        <HeaderMessage>string</HeaderMessage>
        <ClientTracking>string</ClientTracking>
        <TransId>int</TransId>
      </CancelSoapRequestResult>
    </CancelSoapRequestResponse>
  </soap:Body>
</soap:Envelope>
```

4.2.1.4 SOAP 1.2 cancel - response format with sample

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-
envelope">
  <soap12:Body>
    <CancelSoapRequestResponse xmlns="http://tempuri.org/">
      <CancelSoapRequestResult>
        <Successful>string</Successful>
        <ResponseCode>string</ResponseCode>
        <HeaderMessage>string</HeaderMessage>
        <ClientTracking>string</ClientTracking>
        <TransId>int</TransId>
      </CancelSoapRequestResult>
    </CancelSoapRequestResponse>
  </soap12:Body>
</soap12:Envelope>
```

4.2.2 Implementation Examples

4.2.2.1 PHP

Cancel Request using HTTPS POST and cURL

```
$CancelRequest["ClientNumber"] = "000000000";
$CancelRequest["ClientTracking"] = "test";
$CancelRequest["TransID"] = "999";
$CancelRequest["ValidationKey"] = "99999999-F004-4D44-BDB3-E979D6749A64";

// $post_type determines what encoding type – either XML or JSON

if( $post_type == 'xml' ) {
        //XML
        $xmlclass = 'CancelRequest';
        $url1 = "http://www.w3.org/2001/XMLSchema-instance";
        $url2 = "http://www.w3.org/2001/XMLSchema";

        $xml = '<?xml version="1.0" encoding="utf-8"?>';
        $xml .= "<" . $xmlclass . ' xmlns:xsi="' .$url1 .'" ';
        $xml .= ' xmlns:xsd="' .$url2 .'">';

         foreach( $postfields as $x => $y) {
                $xml .= "<$x>".htmlentities( strval($y) ) . "</$x>";
         }

        $xml .= "</" . $xmlclass . ">";
        $curl_data = $xml;

} else {

         // JSON
        $json = json_encode($postfields);
        $curl_data = $json;
}

//Initiate cURL request
$ch = curl_init();

// SureTax Cancel request URL
$suretax_url = 'https://testapi.taxrating.net/Services/V01/SureTax.asmx/CancelPostRequest';

// Set Headers
curl_setopt($ch, CURLOPT_POSTFIELDS, "requestCancel=$curl_data" );  // Assign POST Data
curl_setopt($ch, CURLOPT_POST, 1);                                 // Select POST as transfer type
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0);
curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, 0);
curl_setopt($ch, CURLOPT_URL,$ suretax_url);                       // The Web service URL
curl_setopt($ch, CURLOPT_TIMEOUT, 9);                              // Set timeout


$response = curl_getinfo($ch);
curl_close($ch);

}
```

Note: *No HTTP header should be provided (no need to supply the cURL CURLOPT_HTTPHEADER option.*

Response

```
<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://tempuri.org/">JSON or XML string</string>
```

4.2.2.2 .NET

Cancel Request

```
POST /V01/SureTax.asmx/CancelPostRequest HTTP/1.1
Host: testapi.taxrating.net
Content-Type: application/x-www-form-urlencoded
Content-Length: length
requestCancel=string
```

Response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://tempuri.org/">string</string>
```

5 Appendix A – Implementation Examples

For development and testing, please use the following URL:

https://testapi.taxrating.net/Services/V01/SureTax.asmx (Communications - taxes by invoice)
https://testapi.taxrating.net/Services/V02/SureTax.asmx (Communications- taxes by line item)

or

https://testapi.taxrating.net/Services/V04/SureTax.asmx (General Sales and Communications - Taxes by line item)
https://testapi.taxrating.net/Services/V05/SureTax.asmx (General Sales and Communications - Taxes by invoice)

5.1 PHP

5.1.1 Request using cURL

```
// Web Request Header
$header["ClientNumber"] = "000000001";
$header["BusinessUnit"] = "My Biz Unit";
$header["ValidationKey"] = "12345678-A001-1A11-ABC1-A111A1111A11";
$header["DataYear"] = "2011"
$header["DataMonth"] = "9";
$header["TotalRevenue"] = 100.00;
$header["ReturnFileCode"] = "0";
$header["ClientTracking"] = "1234";
$header["IndustryExemption"] = "";
$header["ResponseGroup"] = "00";
$header["ResponseType"] = "D";

// Input Record 1
$datarecord["LineNumber"] = "1";
$datarecord["InvoiceNumber"] = "112233";
$datarecord["CustomerNumber"] = "000000001";
$datarecord["OrigNumber"] = "2122102100";
$datarecord["TermNumber"] = "2122102100";
$datarecord["BillToNumber"]= "2122102100";
$datarecord["Zipcode"] = "";
$datarecord["Plus4"] = "";
$datarecord["P2PZipcode"] = "";
$datarecord["P2PPlus4"] = "";
$datarecord["TransDate"] = "2011-09-01";
$datarecord["Revenue"] = 100.00;
$datarecord["Units"] = 1;
$datarecord["UnitType"] = "";
$datarecord["Seconds"] = 55;
$datarecord["TaxIncludedCode"] = "0";
$datarecord["TaxSitusRule"] = "01";
$datarecord["TransTypeCode"] = "010101";
$datarecord["SalesTypeCode"] = "R";
$datarecord["RegulatoryCode"] = "01";
$datarecord["TaxExemptionCodeList"] = array( "00");


// $post_type determines what encoding type – either XML or JSON

if( $post_type == 'xml' ) {
         $xmlclass = 'Request';
        $url1 = "http://www.w3.org/2001/XMLSchema-instance";
        $url2 = "http://www.w3.org/2001/XMLSchema";

        $xml = '<?xml version="1.0" encoding="utf-8"?>';
        $xml .= "<" . $xmlclass . ' xmlns:xsi="' .$url1 .'" ';
        $xml .= ' xmlns:xsd="' .$url2 .'">';

         foreach( $header as $x => $y) {
                $xml .= "<$x>".htmlentities( strval($y) ) . "</$x>";
         }

        // Create the items (transactions)
        $xml .= "<ItemList>";

          $xml .= "<Item>";
          foreach( $datarecord as $x => $y) {
                 $xml .= "<$x>".htmlentities( strval($y) );

          if( $x == 'TaxExemptionCodeList' ) {
                          $xml .= "<string>00</string><string>00</string>";
                   }

                 $xml .= "</$x>";
         }
        $xml .= "</Item>";

         $xml .= "</ItemList>";
         $xml .= "</" . $xmlclass . ">";

         $my_data = $xml;

} else {
         // JSON
        $json = json_encode($header);
        $json = substr($json,0,strlen($json)-1);

        $json .= ',"ItemList": [';

        $json .= json_encode( $datarecord );

        $json .= ']}';

        $my_data = $json;
}

// SureTax web request API URL
$suretax_url = "https://testapi.taxrating.net/Services/V01/SureTax.asmx/PostRequest"


//Initiate cURL request
$ch = curl_init();

// Set Headers
curl_setopt($ch, CURLOPT_POSTFIELDS, "request=$my_data" );
curl_setopt($ch, CURLOPT_POST, 1 );
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1 );
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0 );
curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, 0 );
curl_setopt($ch, CURLOPT_URL, $suretax_url );
curl_setopt($ch, CURLOPT_TIMEOUT, 10 );   // 10 second timeout option

$response = curl_exec( $ch );
curl_close( $ch );
```

Note: *No HTTP header should be provided (no need to supply the cURL CURLOPT_HTTPHEADER option.*

Response

```
<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://tempuri.org/">JSON or XML string</string>
```

Note: For JSON implementation, the leading and trailing tags must be removed before calling the PHP **json_decode()** function. This can be accomplished via the following routine: **`$response = substr( substr( $response, 76 ), 0, -9);`**

ALTERNATE PHP with cURL IMPLEMENTATION for JSON:

This alternate cURL implementation for JSON, enables the response to be returned wrapped in a “d” object. For more information on the specifics of this alternative approach, please see the following article:
http://encosia.com/a-breaking-change-between-versions-of-aspnet-ajax/#comment-34045

```
//Using the same initial assignment of the $json variable from the previous example above…

// Add request object to the JSON string
$my_data = '{"request":' ."'";
$my_data.= $json . "'" ;
$my_data.= '}';

// SureTax web request API URL
$suretax_url = "https://testapi.taxrating.net/Services/V01/SureTax.asmx/PostRequest"

//Initiate cURL request
$ch = curl_init();

// Set Headers for JSON
curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type: application/json', 'Content-Length: ' .
strlen($my_data)));

curl_setopt($ch, CURLOPT_POSTFIELDS, $my_data );
curl_setopt($ch, CURLOPT_POST, 1 );
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1 );
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0 );
curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, 0 );
curl_setopt($ch, CURLOPT_URL, $suretax_url );
curl_setopt($ch, CURLOPT_TIMEOUT, 10 );   // 10 second timeout option

$response = curl_exec( $ch );
curl_close( $ch );
```

5.1.2 SOAP request using cURL

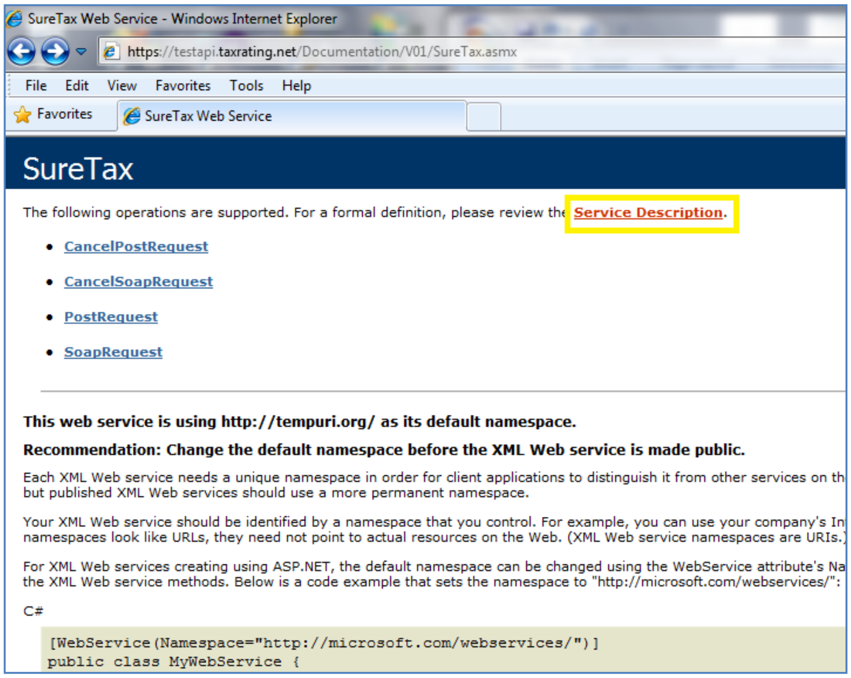
```
// Web Request Header
$header["ClientNumber"] = "000000001";
$header["BusinessUnit"] = "My Biz Unit";
$header["ValidationKey"] = "12345678-A001-1A11-ABC1-A111A1111A11";
$header["DataYear"] = "2011"
$header["DataMonth"] = "9";
$header["TotalRevenue"] = 100.00;
$header["ReturnFileCode"] = "0";
$header["ClientTracking"] = "1234";
$header["IndustryExemption"] = "";
$header["ResponseGroup"] = "00";
$header["ResponseType"] = "D";


// Input Record 1
$datarecord["LineNumber"] = "1";
$datarecord["InvoiceNumber"] = "112233";
$datarecord["CustomerNumber"] = "000000001";
$datarecord["OrigNumber"] = "2122102100";
$datarecord["TermNumber"] = "2122102100";
$datarecord["BillToNumber"]= "2122102100";
$datarecord["Zipcode"] = "";
$datarecord["Plus4"] = "";
$datarecord["P2PZipcode"] = "";
$datarecord["P2PPlus4"] = "";
$datarecord["TransDate"] = "2011-09-01";
$datarecord["Revenue"] = 100.00;
$datarecord["Units"] = 1;
$datarecord["UnitType"] = "";
$datarecord["Seconds"] = 55;
$datarecord["TaxIncludedCode"] = "0";
```

```
$datarecord["TaxSitusRule"] = "01";
$datarecord["TransTypeCode"] = "010101";
$datarecord["SalesTypeCode"] = "R";
$datarecord["RegulatoryCode"] = "01";
$datarecord["TaxExemptionCodeList"] = array( "00");


$url1 = "http://www.w3.org/2001/XMLSchema-instance";
$url2 = "http://www.w3.org/2001/XMLSchema";
$url3 = "http://schemas.xmlsoap.org/soap/envelope/";

$xml = '<?xml version="1.0" encoding="utf-8"?>';
$xml .= '<soap:Envelope' . ' xmlns:xsi="' .$url1 .'" ';
$xml .= ' xmlns:xsd="' .$url2 .'"';
$xml .= ' xmlns:soap="' .$url3 .'">';
$xml .= '<soap:Body>';
$xml .= '<SoapRequest xmlns="http://tempuri.org/">';
$xml .= '<request>';

foreach( $header as $x => $y) {
        $xml .= "<$x>".htmlentities( strval($y) ) . "</$x>";
}

// Create the items (transactions)
$xml .= "<ItemList>";

$xml .= "<Item>";
foreach( $datarecord as $x => $y) {
        $xml .= "<$x>".htmlentities( strval($y) );

        if( $x == 'TaxExemptionCodeList' ) {
                $xml .= "<string>00</string><string>00</string>";
        }

        $xml .= "</$x>";
}
$xml .= "</Item>";

$xml .= "</ItemList>";

foreach( $trailer as $x => $y) {
        $xml .= "<$x>".htmlentities( strval($y) ) . "</$x>";
}

$xml .= '</request></SoapRequest></soap:Body></soap:Envelope>';

$my_data = $xml;

// SureTax web request API URL
$suretax_url = "https://testapi.taxrating.net/Services/V01/SureTax.asmx"

//Initiate cURL request
$ch = curl_init();

// Set cURL Headers
curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type: text/xml; charset=utf-8', 'Content-
Length: '.strlen($my_data), 'SOAPAction: "http://tempuri.org/SoapRequest"' ));

curl_setopt($ch, CURLOPT_POSTFIELDS, $my_data );
curl_setopt($ch, CURLOPT_POST, 1 );
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1 );
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0 );
curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, 0 );
curl_setopt($ch, CURLOPT_URL, $suretax_url );
curl_setopt($ch, CURLOPT_TIMEOUT, 15 );

$response = curl_exec( $ch );
curl_close( $ch );
```

5.2 Microsoft .NET

Follow these steps to configure the SureTax web service:

1. Browse to the documentation API and login using your login credentials (available upon request).
https://testapi.taxrating.net/Documentation/V01/SureTax.asmx


SureTax Web Service - Windows Internet Explorer
https://testapi.taxrating.net/Documentation/V01/SureTax.asmx
File
Edit
View
Favorites
Tools
Help
Favorites
SureTax Web Service

SureTax

The following operations are supported. For a formal definition, please review the **Service Description**.

- CancelPostRequest
- CancelSoapRequest
- PostRequest
- SoapRequest

This web service is using http://tempuri.org/ as its default namespace.

Recommendation: Change the default namespace before the XML Web service is made public.

Each XML Web service needs a unique namespace in order for client applications to distinguish it from other services on th
but published XML Web services should use a more permanent namespace.

Your XML Web service should be identified by a namespace that you control. For example, you can use your company's In
namespaces look like URLs, they need not point to actual resources on the Web. (XML Web service namespaces are URIs.)

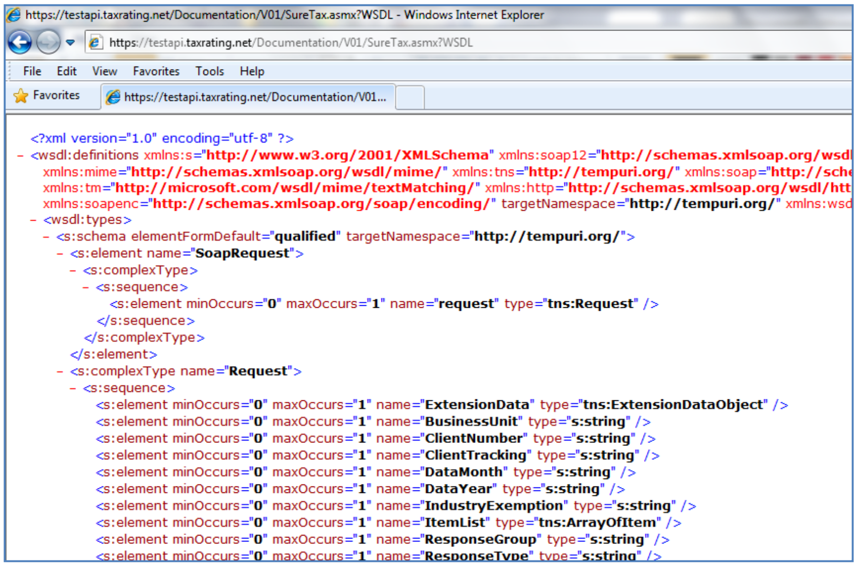
For XML Web services creating using ASP.NET, the default namespace can be changed using the WebService attribute's Na
the XML Web service methods. Below is a code example that sets the namespace to "http://microsoft.com/webservices/":

C#

```
[WebService(Namespace="http://microsoft.com/webservices/")]
public class MyWebService {
```

* WSDL can also be found at: https://testapi.taxrating.net/Documentation/V01/SureTax.asmx?WSDL (requires valid login credentials.)

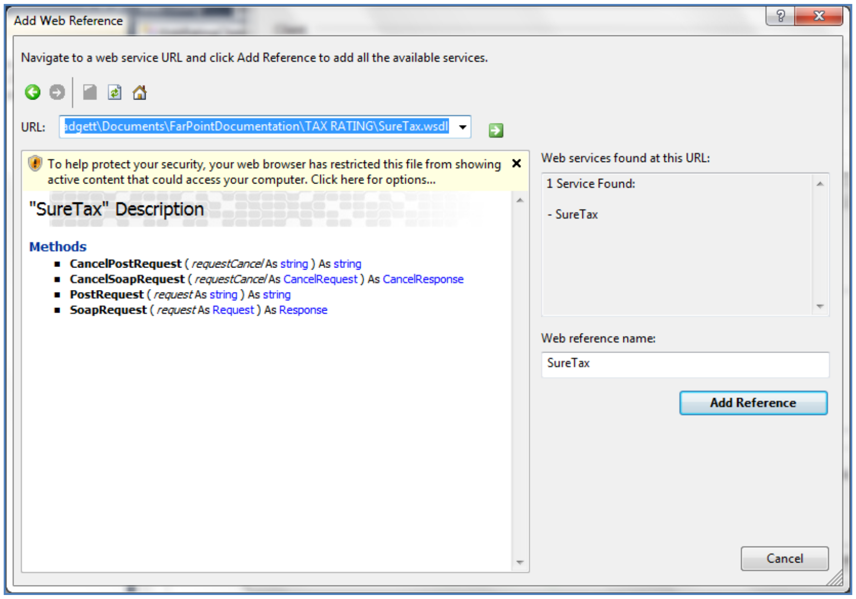
2. Click on the Service Description to display the WSDL:


https://testapi.taxrating.net/Documentation/V01/SureTax.asmx?WSDL - Windows Internet Explorer
https://testapi.taxrating.net/Documentation/V01/SureTax.asmx?WSDL
File
Edit
View
Favorites
Tools
Help
Favorites
https://testapi.taxrating.net/Documentation/V01...

```
  <?xml version="1.0" encoding="utf-8" ?>
- <wsdl:definitions xmlns:s="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://schemas.xmlsoap.org/wsdl
    xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:tns="http://tempuri.org/" xmlns:soap="http://sche
    xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/" xmlns:http="http://schemas.xmlsoap.org/wsdl/htt
    xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" targetNamespace="http://tempuri.org/" xmlns:wsd
  - <wsdl:types>
    - <s:schema elementFormDefault="qualified" targetNamespace="http://tempuri.org/">
      - <s:element name="SoapRequest">
        - <s:complexType>
          - <s:sequence>
              <s:element minOccurs="0" maxOccurs="1" name="request" type="tns:Request" />
            </s:sequence>
          </s:complexType>
        </s:element>
      - <s:complexType name="Request">
        - <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="ExtensionData" type="tns:ExtensionDataObject" />
            <s:element minOccurs="0" maxOccurs="1" name="BusinessUnit" type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="ClientNumber" type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="ClientTracking" type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="DataMonth" type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="DataYear" type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="IndustryExemption" type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="ItemList" type="tns:ArrayOfItem" />
            <s:element minOccurs="0" maxOccurs="1" name="ResponseGroup" type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="ResponseType" type="s:string" />
```

3. Right click on the WSDL and select view source. Save the source file as SureTax.wsdl.

4. In Visual Studio, select "Add Web Reference" and paste the path to the SureTax.wsdl you saved earlier in for the URL. Name the web reference SureTax:



Navigate to a web service URL and click Add Reference to add all the available services.

URL: edgett\Documents\FarPointDocumentation\TAX RATING\SureTax.wsdl

To help protect your security, your web browser has restricted this file from showing active content that could access your computer. Click here for options...

"SureTax" Description

Methods

- **CancelPostRequest** (*requestCancel* As string) As string
- **CancelSoapRequest** (*requestCancel* As CancelRequest) As CancelResponse
- **PostRequest** (*request* As string) As string
- **SoapRequest** (*request* As Request) As Response

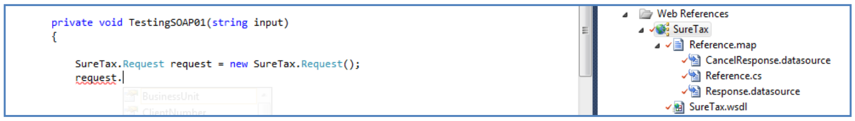
Web services found at this URL:

1 Service Found:

- SureTax

Web reference name: SureTax

5. The SureTax Web Service should now be available for use in the code:



```
private void TestingSOAP01(string input)
{

    SureTax.Request request = new SureTax.Request();
    request.
```

6. To run the web service, you must change the connection setting in the **web.config** file by updating entry from "Documentation" to "Services":

From:

```
<applicationSettings>
  <WebRatingClient.Properties.Settings>
    <setting name="WebRatingClient_SureTax_SureTax" serializeAs="String">
       <value>https://testapi.taxrating.net/Documentation/V01/SureTax.asmx</value>
    </setting>
  </WebRatingClient.Properties.Settings>
</applicationSettings>
```

To:

```
<value>https://testapi.taxrating.net/Services/V01/SureTax.asmx</value>
```

5.2.1 .NET Examples

Request

```
POST URL Services/V01/SureTax.asmx/PostRequest HTTP/1.1
Host: testapi.taxrating.net
Content-Type: application/x-www-form-urlencoded
Content-Length: length
request=string
```

Response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://tempuri.org/">string</string>
```

SOAP format with sample data

This sample assumes a web reference has been established.

```
//Create and populate request
Request request = new Request();
request.BusinessUnit = "BizUnit";
request.ClientNumber = "000007";
request.ClientTracking = "TrackThis";
request.DataMonth = "9";
request.DataYear = "2011";
request.IndustryExemption = "";
request.ResponseType = "S";
request.ResponseGroup = "00";
request.ReturnFileCode = "0";
request.TotalRevenue = 40.00M;
request.ValidationKey = "ABC123DE-12AB-23CD-34EF-9876ZYXW1234";
//Create and populate item
Item item = new Item();
item.BillToNumber = "8585260000";
item.CustomerNumber = "01";
item.InvoiceNumber = "1";
item.LineNumber = "1";
item.OrigNumber = "8585260000";
item.P2PPlus4 = string.Empty;
item.P2PZipcode = string.Empty;
item.Plus4 = string.Empty;
item.RegulatoryCode = "99";
item.Revenue = 40.00M;
item.SalesTypeCode = "R";
item.Seconds = 55;
item.TaxExemptionCodeList = new string[0];
item.TaxIncludedCode = "0";
item.TaxSitusOverrideCode = "";
item.TaxSitusRule = "01";
item.TermNumber = "8585260000";
item.TransTypeCode = "010101";
item.TransDate = "1/1/2000";
item.Units = 1;
item.UnitType = "00";
item.Zipcode = string.Empty;
//Add item to request
request.ItemList = new Item[1];
request.ItemList[0] = item;
//Create service proxy and call Process method.  Capture call results
SureTax rateInvoice = new SureTax();
Response response = rateInvoice.SoapRequest(request);
//Access response results
foreach (var group in response.GroupList)
    foreach (var tax in group.TaxList)
    {
        string taxAmount = tax.TaxAmount;
        string taxTypeCode = tax.TaxTypeCode;
        string taxTypeDesc = tax.TaxTypeDesc;
    }
```

5.3 Using GZIP Compression

SureTax supports GZIP compression for web requests and responses. For requests that have a large number of items (transactions), the use of GZIP compression can be very advantageous as it reduces the size of the payload being transmitted. The performance impact of the compression activities is negligible.

To enable GZIP compression with SureTax, the following header attribute must be submitted in the web request:
"Accept-Encoding: gzipped"

See Section 5.1 for detailed implementation examples.

.NET Example

```
var post = "request=< ADD XML HERE > ";

//gzip the POST data
var bigRequestStream = new MemoryStream(Encoding.UTF8.GetBytes(post));
var tinyRequestStream = new System.IO.MemoryStream();
using (var gzipStream = new GZipStream(tinyRequestStream, CompressionMode.Compress, true)) {
bigRequestStream.CopyTo(gzipStream); }
tinyRequestStream.Position = 0;

//create the request to SureTax
var request =
(HttpWebRequest)WebRequest.Create("https://testapi.taxrating.net/Services/V02/SureTax.asmx/PostRequest");
request.Method = "POST";
request.ContentType = "application/x-www-form-urlencoded";
request.ContentLength = tinyRequestStream.Length;
//Add gzip indicator
request.Headers.Add(HttpRequestHeader.AcceptEncoding, "gzipped");

//add gzip POST data
Stream dataStream = request.GetRequestStream();
tinyRequestStream.WriteTo(dataStream);
dataStream.Close();

//submit, receive gzip response and decompress
var tinyResponseStream = ((HttpWebResponse)request.GetResponse()).GetResponseStream();
var bigResponseStream = new System.IO.MemoryStream();
using (var gzipStream = new GZipStream(tinyResponseStream, CompressionMode.Decompress, true)) {
gzipStream.CopyTo(bigResponseStream); }

//display decompressed response to validate
bigResponseStream.Position = 0;
Console.WriteLine(new StreamReader(bigResponseStream).ReadToEnd());
```

PHP Example

```
// SureTax web request API URL
$suretax_url = "https://testapi.taxrating.net/Services/V02/SureTax.asmx/PostRequest"

//Initiate cURL request
$ch = curl_init();

$my_data = "*** JSON STRING ***";

// Set Headers
curl_setopt($ch, CURLOPT_HTTPHEADER, array('Accept-Encoding: gzipped') );
curl_setopt($ch, CURLOPT_ENCODING , 'gzip');
$post_string = gzencode("request=$my_data");
curl_setopt($ch, CURLOPT_POSTFIELDS, $post_string );

curl_setopt($ch, CURLOPT_POST, 1 );
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1 );
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0 );
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 0);
curl_setopt($ch, CURLOPT_URL, $suretax_url );
curl_setopt($ch, CURLOPT_TIMEOUT, 10 );

$response = curl_exec( $ch );
curl_close( $ch );

// Uncompress GZIP response
$response = gzinflate(substr($response,10,-8));  // Note: Or use gzdecode() with PHP v5.4 or greater
```

6 Appendix B – Transaction Type Codes

The most current SureTax Transaction Type Code list including their CCH ZipComm Group/Item corresponding values is available as an addendum to this API document. For additional information on these codes and their specific definitions, please see the Telecom Mapping Guide.

7 Appendix C – Provider Type (Regulatory) Codes

The following are the list of Provider Type (Regulatory) codes used in the calculation process.

Type	Regulatory Code	Description
Default	99	Default value for non-telecom services or if Provider Type is unknown.
Communications	00	ILEC
	01	IXC
	02	CLEC
	03	VoIP
	04	ISP
	05	Wireless
Utilities	50	Regulated Distributor
	52	Regulated Distributor/Supplier
	53	Non-Regulated Reseller
	54	Non-Regulated Supplier With a Regulated Distributor
	55	Custom - Utilities
	56	Regulated Pipeline Company
	57	Renewable Energy
Retail	70	Retail

8 Appendix D – Tax Exemption Record Indicators

Exemption Indicator	Exemption Description	Tax Type(s) Exempted	Tax Authority Levels Exempted	Detailed Description
00	No Exemptions	None	n/a	No tax exemptions apply
01	Federal Excise Tax Exempt	00	0 – Federal	
02	State Taxes Exempt	All State	1 – State	
03	Federal Excise Tax and State Taxes Exempt	00 All State	1 – State	
04	Local Taxes Exempt	All	2 – County 3 – City 4 – Local 5 – Reporting Agency	All tax types are exempted for the "local" tax authority levels
05	Federal Excise Tax and Local Taxes Exempt	00 All Local	2 – County 3 – City 4 – Local	
06	State and Local Taxes Exempt	All	1 – State 2 – County 3 – City 4 – Local 5 – Reporting Agency	
07	Federal Excise Tax and State and Local Taxes Exempt	00 All State All Local	1 – State 2 – County 3 – City 4 – Local 5 – Reporting Agency	
09	All Federal Level Taxes Exempt	00 35 60	0 – Federal	All tax types are exempted for the "Federal" tax authority (i.e. FET, FUSF, FTRS)
10	All Unit Based Taxes Exempt	See note	See note	This exemption type excludes all tax types at all tax authority levels that are listed as a unit based fee.
11	Unit based E911 Exempt	06 33	Any Level	Exempts state and local unit based E911 fees.
12	Only apply Unit Based Fees	See note	See note	This exemption type excludes all tax types at all tax authority levels that are not listed as a unit based fee.
13	Local Right of Way Fees (ROW) Exempt	24	2 – County 3 – City 4 – Local	
14	All State and Local E911s Exempt	06 33	Any Level	Exempts state and local E911 fees - both unit based and percentage of revenue.
19	Only Apply Federal Universal Service Fund (USF)	See note	See note	All taxes except Federal USF (tax type 35) are exempt
20	All PUC Fees Exempt	08	Any Level	
21	All State and Local Gross Receipts Tax (GRT) Exempt	28 32	Any Level	
22	All State Universal Service Fund (USF) Fees Exempt	26	Any Level	
23	State Gross Receipts Tax and Universal Service Fund Exempt	26 28 32	Any Level	

24	Federal Universal Service Fund (USF) Exempt	35	0 – Federal	
25	Federal Excise, Federal Universal Service Fund (USF) and State Sales Tax Exempt	00 01 35	Any Level	
26	State Excise Tax Exempt (includes FL CST)	27	1 – State	
27	Federal Excise, Federal USF and Local Sales Tax Exempt	00 02 03 04 05 35	0 – Federal 2 – County 3 – City 4 – Local 5 – Reporting Agency	
28	Federal USF and State Local Sales Tax Exempt	01 02 03 04 05 35	Any Level	
29	Federal USF and Local Sales Tax Exempt	02 03 04 05 35	0 – Federal 2 – County 3 – City 4 – Local 5 – Reporting Agency	
30	State Excise and Local Telecommunications Tax Exempt	27 37	1 – State 2 – County 3 – City 4 – Local	This is primarily used for exempting the Florida State and Local Telecommunications Services Tax.
31	State Level Sales Tax Exempt	01	1 – State	
32	State and Local Sales Tax Exempt	01 02 03 04 05	Any Level	
33	Federal Excise, Federal USF, State and Local Sales Tax Exempt	00 01 02 03 04 05 35	0 – Federal 1 – State 2 – County 3 – City 4 – Local 5 – Reporting Agency	
35	Federal TRS (Federal Regulatory Assessment Fee) Exempt	60	0 – Federal	
36	Value Added Tax (VAT)	51	Any level	This for any international VAT transaction (not including Canada as they have separate tax types).
37	Canadian Provincial Service Taxes (PST) Exempt	42	Any Level	
38	Canadian Harmonized Service Taxes (HST) Exempt	41	Any Level	
39	Canadian Goods and Services Taxes (GST) Exempt	40	Any Level	
41	Local Utility Users Tax Exempt	16	Any Level	

42	State Excise Tax Exempt and Local Surcharge Exempt	17 27	1 – State 4 – Local	This is primarily used for exempting the NY State Excise and local MTA Surcharge on Excise tax.
43	Universal Lifeline Telephone Service Surcharge (ULTS) exempt	22	Any Level	
44	City Level Taxes Exempt	All	3 – City	
45	State Franchise Fee	23	1 – State	
46	Local Franchise Fee	36	2 – County 3 – City 4 – Local 5 – Reporting Agency	
47	County Level Taxes Exempt	All	2 – County	
61	Los Angeles Communications Users Tax Exempt	61	See Note	Only for Geocode starting with "US06037", which is all localities in Los Angeles that apply the CUT.
62	Non-unit taxes exempt except for E911	Many	Any Level	Used to disable all revenue based taxes with the exception of E911. In addition to unit based fees, only state-level, revenue based E911 would apply.
97	*Reserved*			
98	*Reserved*			
99	All Taxes Exempt – Apply no tax or fees	All	All	All tax types for all levels are exempted.

8.1 Specialty Exemptions

In addition to the Tax Exemptions provided above, SureTax provides for specialty customer exemptions. These exemptions are provided from the CCH Customer Exemptions Smart Charts and will disable all taxes for customers belonging to one of these classifications.

Like the Tax Exemption Record Indicators, these codes are submitted to SureTax as part of the request in the TaxExemptionCodeList field.

Please contact SureTax for information about these codes.

9 Appendix E – Unit Type Codes

The following are the list of unit type codes used in the calculation process.

Type	Unit Type Code	Description
Default	00	Default value for non-telecom services.
Communications	00	Access Line
Utilities	03	kWh
	04	THERM
	06	Meter
	07	BCF (1 Billion Cubic Feet)
	08	dTh (dekatherm
	09	mWh (1000 kWh)
	10	kW (Kilowatts)
	11	mW (Megawatts)
	12	CCF (100 Cubic Feet)
	14	MCF (1000 Cubic Feet)
	18	Canadian THERMS
	21	Per Liter
	22	Per Gallon

10 Appendix F – Tax Type Codes

The following are the list of tax type codes used in the calculation processes. These codes will be displayed on the tax reports and outputs generated from the system. Please note the first digit actually refers to the tax "level" - fed, state, etc. See list provided in Appendix G.

Tax Type Code	Tax Type Description	Tax Authority
000	Federal Excise Tax	Federal
035	Federal Universal Service Fund	Federal
040	Goods And Services Tax (GST)	Federal
051	Value Added Tax (VAT)	Federal
059	FUSF Recovery	Federal
060	Federal TRS Fund	Federal
101	State Sales Tax	State
106	State 911 Tax	State
107	Poison Control	State
108	PUC Fee	State
109	Telecomm Relay Systems Surcharge	State
110	State License Tax	State
111	State Consumption Tax	State
117	Misc. Surcharge 1	State
118	Misc. Surcharge 2	State
119	Misc. Surcharge 3	State
120	Misc. Surcharge 4	State
122	Universal Lifeline Telephone Service Surcharge	State
123	State Franchise Fee	State
124	Local Right-Of-Way Fee	State
126	Universal Service Fund	State
127	State Excise Tax	State
128	State Gross Receipts Tax	State
129	State Infrastructure Maintenance Fee	State
133	Local 911 Surcharge	State
142	Provincial Sales Tax (PST)	State
144	State Lease/Rental Tax	State
149	State Utility Tax	State
150	State Business And Occupation Tax	State
202	County Sales Tax	County
203	County Local Sales Tax	County
204	City Sales Tax	County
216	Local Utility Users Tax	County
224	Local Right-Of-Way Fee	County
233	Local 911 Surcharge	County
237	Local Telecommunications Tax	County

238	Local License Tax	County
245	County Lease/Rental Tax	County
302	County Sales Tax	City
304	City Sales Tax	City
305	City Local Sales Tax	City
316	Local Utility Users Tax	City
324	Local Right-Of-Way Fee	City
331	Local Business And Occupation Tax	City
332	Local Gross Receipts Tax	City
333	Local 911 Surcharge	City
336	Local Franchise Fee	City
337	Local Telecommunications Tax	City
338	Local License Tax	City
343	Local Franchise Agreement	City
347	City Lease/Rental Tax	City
348	City Local Lease/Rental Tax	City
403	County Local Sales Tax	Local
404	City Sales Tax	Local
405	City Local Sales Tax	Local
416	Local Utility Users Tax	Local
417	Misc. Surcharge 1	Local
418	Misc. Surcharge 2	Local
433	Local 911 Surcharge	Local
5C2	User defined Pass-through Tax	Reporting Agency
533	Local 911 Surcharge	Reporting Agency

11 Appendix G – Tax Authority Level Codes

The following are the list of tax type authority codes used in the calculation processes. These codes will be displayed on the tax reports and outputs generated from the system. These values are the same as the first character of the Tax Type Codes listed in Appendix C.

Tax Authority Code	Tax Authority Description
0	Federal
1	State
2	County
3	City
4	Local
5	Reporting Agency

12 Appendix H – Response Code and Messages

The following are the list of message codes used when a Web Request fails. Web Requests will fail on first failure detected and will return the associated message.

Note: *Before specific data elements of a web request are validated, the SureTax serializer checks for proper formatting / encoding across the entire request. Should an error occur in this process, an* ***1101*** *error code will result with a header message such as "Failure - There was an error deserializing the object of type…".*

12.1 Header Level Response Codes

Message Code	Message Code Description
1000	Failure – STAN already processed
1100	Failure – General Failure
1101	Failure – There is a format error in the web request - Input string was not in a correct format.
1110	Failure – Data Month Required
1120	Failure – Data Year Required
1121	Failure – Invalid Data Year/Month – Must be a published Data Year/Month
1130	Failure – Client Number Required
1131	Failure – Invalid Client Number
1141	Failure – Invalid Business Unit
1150	Failure – Validation Key Required
1151	Failure – Invalid Validation Key
1171	Failure – Invalid Client Tracking Code
1190	Failure – Response Group Code Required
1191	Failure – Invalid Response Group Code
1200	Failure – Response Type Required
1201	Failure – Invalid Response Type
1210	Failure – Return File Code Required
1211	Failure – Invalid Return File Code
1220	Failure – Item List Required
1510	Failure - Transaction is more than 60 days old (generated from ***Cancel Request*** method)
9001	Success with Item Errors (see table 9.2)
9999	Success

12.2 Item Level Response Codes

Message Code	Message Code Description
9100	Invalid Invoice Number – Must be Alphanumeric
9101	Invalid Invoice Number – Exceeds Character Limit (20)
9110	Invalid Customer Number – Must be Numeric
9111	Invalid Customer Number – Exceeds Character Limit (10)
9120	Invalid Bill To Number – Must be a 10 digit number
9121	Invalid Bill To Number – NPANXX not found
9130	Invalid Origination Number – Must be a 10 digit number
9131	Invalid Origination Number – NPANXX not found
9140	Invalid Termination Number – Must be a 10 digit number
9141	Invalid Termination Number – NPANXX not found
9150	Invalid Billing Zip Code = US – Must be a 5 digit number
9151	Invalid Billing Zip Code or P2P Zip code [optional additional error information]
9152	Invalid Billing Zip Code or P2P Zip code [optional additional error information]
9153	Invalid Country Code – Must be 2 characters
9155	Invalid Geocode – cannot be null

9160	Invalid Billing Zip Code Extension – Must be a 4 digit number
9170	Invalid Secondary Zip Code – cannot be null
9171	Invalid Secondary Zip Code = CAN – Must be a 6 character Alpha-Numeric
9172	Invalid Secondary Zip Code = Zip Code not found
9180	Invalid Secondary Zip Code Extension – Must be a 4 digit number
9190	Invalid Date of Call – Format Does Not Match a Valid US Date Format
9200	Invalid Line Number – Exceeds Character Limit (20)
9220	Invalid Unit Type – cannot be null
9240	Invalid Tax Included Code – cannot be null
9250	Invalid Tax Situs Rule Code
9260	Invalid Transaction Type Code
9270	Invalid Sales Type Code – cannot be null
9280	Invalid Regulatory Code – cannot be null
9290	AuxRevenueType Required
9410	Transaction is already cancelled (generated from ***Cancel Request*** method)

13 Appendix I – Deprecated Features

These deprecated features can still be used, but not recommended given alternatives available.

13.1 Alternate Engine (V03)

The “Alternate” Engine V03 that was supported in API version 1.2 (and lastly in version 1.42) has been replaced with Engine V02.

The Request input values for Engine V03 are the same as Engine V02, but the V02 response, contains three additional values:

FeeRate	The unit based fee for the tax type in format $.CCCC (if zero, please see TaxRate)
RevenueBase	The effective revenue for the tax provided in the TaxAmount field. This amount can be different than the amount in the Revenue field when taxes are impacted by specific exemptions and/or tax on tax.
TaxOnTax	The amount of tax on tax attributed to the final amount of tax. Please note this amount is included in the TaxAmount field total and is provided here separately only for reference purposes.

While the V03 Engine will continue to be supported for backward compatibility, it is recommended that V02 Engine is used for all new communications implementations that require taxes by line item.

14 Appendix J – Change History

Date	Comments
2/1/2011	First draft.
9/30/2011	V 1.0.1
12/23/2011	V1.1 - Added Cancel Request methods
1/12/2012	V1.11 - Added max field lengths to request layout - Misc corrections to examples
2/16/2012	V1.12 - Added Transaction Type codes for: Shipping, Prepaid Wireless Retailer, Information Services, Security Monitoring, and Misc Telecom – Returned Check Fee - Corrected WSDL - Added example for .NET Web Service setup
4/17/2012	V1.13 - Added Private Line Transaction Types
6/27/2012	V1.14 - Updated Private Line Transaction Types - Updated Leasing Transaction Types - Updated required / optional field designations - Updated Request ordering in examples - Added TransType 980413 for short term Lease
8/3/2012	V1.15 - Added TransType codes for Fax over IP (both Fixed and Nomadic) - Added Tax Exempt TransType code 999999 - Updated various examples
11/2/2012	V1.2 - Updated ResponseType input field to support decimal precision - Added information and response examples for the Alternate Processing Engine (V03) - Added example for alternate object type response for HTTPS/POST JSON - Updated various examples - Cable Services Transaction Type codes added
1/28/2013	V1.21 - Added New TransType codes for VoIP SIP Trunks (both Fixed and Nomadic) - Added new Exempt Type codes
3/29/2013	V1.22 - Added New TransType codes for Data Lines - Added new Sales type codes for Industrial and Lifeline - Added and updated Tax Exemption Indicators descriptions - Added Specialty Tax Exemption indicators
5/31/2013	V1.23 - Corrected Cancel Request example - Added XML serialization and decoding information - Added new TransType codes for Non-Interconnected VoIP - Added new Exempt Type for local ROW fees
7/25/2013	V1.24 - Expanded Header->Client Tracking Field to a maximum of 100 characters - Added Computer and Software Services Transaction Type codes - Added EDI Transaction Type codes - Changed Exemption code “36” to “39” - Added new Exemption codes “35” and “36” - Added new tax type codes for VAT and a user-defined code
8/16/2013	V.1.24.1 - Added additional information on JSON encoding requirements

11/5/2013	V.1.24.2 - Removed reference to V03 engine - Added new information on V02 engine - Added Tax Situs Rule 11 information
7/21/2014	V.2.0 - Expired V03 Engine - Added new Engines – V04 and V05 -
12/5/2014	V.2.1 - Added additional tax exemption indicators - Added tax situs rule “17”information
12/18/2014	V.2.2 - Added utilities information – unit types and provider types - Corrected error messages listing - Various corrections
1/16/2015	V.2.21 - Added GZIP compression information and examples - Various corrections to PHP examples
8/17/2015	V.2.22 - Expanded customer number and invoice number to a length of 40 - Expanded line number to a length of 40 - Updated API error messages

GENERAL LEGAL DISCLAIMER

WHILE WE ARE PLEASED TO PROVIDE THE MATERIALS THAT YOU REQUESTED, WE MUST MAKE CLEAR THAT WE ARE PROVIDING THESE MATERIALS TO ASSIST YOU IN YOUR OWN RESEARCH AND ANALYSIS RELATING TO THE SUBJECT MATTER OF THE RESEARCH. IN THIS REGARD, OUR PROVIDING THESE MATERIALS IS CONDITIONED UPON THE FOLLOWING TERMS AND YOUR RELIANCE UPON OR USE IN ANY RESPECT OF THE MATERIALS CONFIRMS YOUR ACCEPTANCE OF THESE TERMS:

THE CONTENTS OF THIS PUBLICATION ARE BELIEVED TO BE ACCURATE. HOWEVER, RESPONSIBILITY CANNOT BE ASSUMED FOR THE INFORMATION CONTAINED HEREIN, AND THE CONSEQUENCES RESULTING FROM THE USE THEREOF. MATERIAL IN THIS PUBLICATION IS SUBJECT TO CHANGE WITHOUT NOTICE. THIS DOCUMENTATION AND THE COMPUTER SOFTWARE IT DESCRIBES ARE DESIGNED TO PROVIDE ACCURATE AND AUTHORITATIVE INFORMATION IN REGARD TO THE SUBJECT MATTER COVERED. THEY ARE DISTRIBUTED WITH THE UNDERSTANDING THAT THE PUBLISHER IS NOT ENGAGED IN RENDERING ACCOUNTING OR OTHER PROFESSIONAL SERVICE. IF LEGAL ADVICE OR OTHER EXPERT ASSISTANCE IS REQUIRED, THE SERVICES OF A COMPETENT PROFESSIONAL SHOULD BE SOUGHT.

THE MATERIALS ARE BEING PROVIDED TO YOU “AS IS” WITHOUT WARRANTY OF ANY KIND. CCH EXPRESSLY DISCLAIMS ALL WARRANTIES RELATING TO THE MATERIALS, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

CCH AND ANY OF ITS AFFILIATES IS NOT ENGAGED IN RENDERING LEGAL, ACCOUNTING, TAX OR OTHER PROFESSIONAL SERVICES. IF LEGAL, ACCOUNTING, TAX OR OTHER EXPERT ASSISTANCE IS REQUIRED; THE SERVICES OF A COMPETENT PROFESSIONAL SHOULD BE OBTAINED.

YOU ASSUME ALL RESPONSIBILITIES AND OBLIGATIONS WITH RESPECT TO ANY DECISIONS OR ADVICE MADE OR GIVEN AS A RESULT OF THE USE OF THE MATERIALS BEING PROVIDED. YOU EXPRESSLY AGREE THAT USE OF THE MATERIAL IS AT YOUR OWN RISK AND THAT THE MATERIALS ARE NOT INTENDED TO REPLACE YOUR PROFESSIONAL SKILL AND JUDGMENT.

NEITHER CCH NOR ANY OF ITS AFFILIATES SHALL HAVE ANY LIABILITY TO YOU OR ANYONE ELSE FOR ANY INACCURACY, ERROR OR OMMISSION CONTAINED IN THE MATERIALS OR ARISING FROM YOUR USE IN ANY RESPECT OF THE MATERIALS, REGARDLESS OF THE CAUSE. THE FOREGOING EXCLUSION OF LIABILITY IS INTENDED TO APPLY REGARDLESS OF THE FORM OF CLAIM.

THE RESEARCH MATERIALS BEING PROVIDED RELATE TO PRODUCTS AND/OR SERVICES ACQUIRED BY YOU PURSUANT TO OTHER AGREEMENTS BETWEEN CCH AND YOU. THE PROVISION OF THESE RESEARCH MATERIALS IS NOT INTENDED TO CHANGE OR OTHERWISE AFFECT, AND IN NO WAY CHANGES OR OTHERWISE AFFECTS, THE RIGHTS AND OBLIGATIONS UNDER THOSE AGREEMENTS; YOUR USE OF SUCH PRODUCTS AND/OR SERVICES SHALL CONTINUE TO BE GOVERNED BY THE TERMS OF THOSE AGREEMENTS.

FINALLY, THE MATERIALS BEING PROVIDED ARE CONFIDENTIAL AND ARE BEING PROVIDED TO YOU FOR YOUR PERSONAL AND NON-COMMERCIAL USE. YOU MAY NOT DISTRIBUTE ANY PORTION OF THE MATERIALS OUTSIDE OF YOUR COMPANY WITHOUT OUR PRIOR WRITTEN AUTHORIZATION.

Copyright 2015 CCH INCORPORATED. A Wolters Kluwer business. All Rights Reserved Worldwide. Material in this publication may not be reproduced or transmitted in any for or by any means without pior writtern permission.