AvaTax for Communications

SaaS Pro Telecom Developer Manual

Release:	9.18.1712.1
Document:	TM_00116_0042
Date:	11/22/2017

Avalara for Communications - Contact Information	
Address	Avalara, Inc.
	513 South Mangum Street, Suite 100
	Durham, North Carolina, 27701
Toll Free	800-525-8175
Corporate Site	http://communications.avalara.com/
Comms Platform Site	https://communications.avalara.net
Email	communicationsupport@avalara.com

Document Revision History

The Revision History log lists the date and description of the most recent revisions or versions of the document.

Date	Version	Description
02/19/2016	0026	Avalara branding updates to reflect the transition to the new company and product names have been incorporated into this document. Please see Appendix A – Avalara Product Names for specific changes in product references and descriptions.
05/18/2016	0027	Updates to support Custom Log and Category Exemptions include: Addition of new **Section 1.8 Exemptions**, updates to **Section 1.15 Proration**; updates to **Section 3 Customer Mode**, added new **Section 4 Generating Reports**, updates to **Section 7 Utility Web Methods** (CreateReport) and updates to **Section 8 Web Service Data Definitions** (ReportOptions, CustomLogField and CategoryExemption tables)
06/17/2016	0028	Added new **Section 1.16 Tax Grouping**, **Section 3.6 Supported APIs** for Customer Mode, **Section 5 Bridge Conferencing**, and new tables added to **Section 9 Web Service Data Definitions** to support bridge conferencing. Added note to **Section 7.24 CalcJurisdiction** and updated **Section 2** with details for entry of Canadian postal codes.
08/02/2016	0029	Added note in **Section 1.7 Exclusions** regarding excluded tax jurisdictions appearing as unknown or not appearing on TSRs. Added new **Section 6 Optional Fields** to describe default and extended optional fields. Updated **Section 8 Utility Web Methods** with new APIs, **GetOptionalFieldKeyDesc** and **UpdateOptionalFieldKeyDesc** to support additional alphanumeric field for AFC SaaS Pro. OptionalField added to Transaction table and an OptionalField and OptionalKey table added in **Section 9 Web Service Data Definitions**.
09/26/2016	0030	Updated **Section 4.4 Custom Log Report Columns** to include columns for the Extended Optional fields. Added new **Section 12 Monthly Update Procedures** for downloading documentation.
11/15/2016	0031	Updated description in **Section 1.9.2 Level Exemptions**. Updated single transaction maximum from 3,000 to 50,000 in **Section 3 Customer Mode**. Added new Section **3.7 Tax Inclusive Transactions** to indicate how to process tax inclusive calculations in customer mode. Added new **Section 7 Zip Lookup Requests** and **11.16 Zip Lookup** to provide an overview of zip lookup functionality and use of the API. Added new **Section 10.21 ProcessCustomerBatchV2** API and **Section 10.23 CalcTaxesInCustModeV2** API. Added new **Section 12 REST Interface APIs**. Updated **Section 13 Web Data Definitions** includes updated data structure tables for **Transaction**, and new ones for **TaxDataV2**, **CustomerResultsV2**, **ZipLookup**, **ZipLookupResults** and **LocationData**. Also, updated the **BridgeConferenceParticipantResult** table in **Section 13** to show the updated value for the errorcode field.
12/05/2016	0032	Updated **Section 3 Customer Mode** with note providing recommendation on appropriate duration of a transaction in Invoice Mode. Updated **Section 4.6.1 Accessing the FTP Site** to reflect utilization of a secure file-transfer protocol (SFTP) in addition to FTP. Updated remarks in **Section 11.7 ZipToPCode.** Added **Section 8 Safe Harbor Overrides for Traffic Studies** and updated **Section 13 Web Data Definitions** with the new **SafeHarborOverride** data structure table.
01/11/2017	0033	Updated **Section 1.14 Private Line** to reflect support of additional APIs. Also updated copyright year on document.

Date	Version	Description
01/18/2017	0034	Added notes to **Section 1.13 Interstate/Intrastate Determination** and **Section 1.14 Private Line**. Also updated tables in **Section 3.6 Supported and Non-Supported Features** (under Customer Mode).
02/13/2017	0035	Added new **Section 1.1 Getting Started** which includes requirements for integration and key inputs. Updated **Section 1.4 Transaction Date** with additional details for date format.
03/13/2017	0036	All references to 'reverse' tax calculations in AFC SaaS Pro have been updated and renamed to reflect the current naming convention which is 'tax inclusive' calculations. As a result, 8 APIs in **Section 10 SOAP Telecom Web Methods** have been updated and renamed. The table in **Section 12 in REST Interface APIs** has also been updated. Updated **Section 1.9.3 Tax type exemptions** with note regarding availability of wildcard character for tax types. Release number and version updates. Updated **Section 1.15 Private Line** to include expanded support of the feature with adjustment APIs. Added new **Section 1.18 Specifying a Unique Identifier**. Updated description in **Discount Table** in **Section 14 Web Service Enumeration Definitions.**
04/14/2017	0037	Added new **Section 1.19 Commit/Uncommit** and **Section 1.20 Application of Tax Brackets and Limits**. All references to 'Customer Mode' in AFC SaaS Pro have been updated and renamed to reflect the current naming convention which is 'Invoice Mode.' As a result, **Section 3** has been updated and renamed **Invoice Mode**. Added new error message to **Section 10 Error Messages Common to all Calculation Methods**.
05/25/2017	0038	Updated **Section 1.3 Client Information**. Added **Section 10.39** for **CommitTransactions** API. Added **CommitTransactions** to the table in **Section 10 SOAP Telecom Web Methods** and to the table in **Section 12 REST Interface APIs**. Added **CommitData** data structure table to **Section 13 Web Service Data Definitions.**
06/27/2017	0039	Updated links in **Section 1.1 Getting Started**. Updated **Section 1.14 Interstate/Intrastate** and **Section 1.15 Private Line** to include Invoice Mode APIs. Added **Section 1.21 Use of Client IDs in the REST Interface**. Updates throughout **Section 12 REST Interface APIs** and **Section 15 Monthly Update**.
07/07/2017	0040	Miscellaneous updates in **Section 3 Invoice Mode**.
10/25/2017	0041	Updated Avalara contact information (address and support site). Removed **Appendix A – Avalara Product Names**.
11/22/2017	0042	Added note related to timestamp/invoice date passed in **Section 1.4 Transaction Date**. Updated **Section 2 Specifying a Tax Jurisdiction** with a note and table providing overview of alternate formats for US zip code plus 4 digits. Updated **Section 10.31 CalcTaxesInCustModeV2** to include **SafeHarborOverride** as a parameter. Added private line fields to the Transaction table in **Section 13 Web Service Data Definitions**. Added new REST APIs to the table in **Section 12 REST Interface APIs**. Updated **Section 14 Web Service Enumeration Definitions** with an **ExemptionType** table which provides values and descriptions.

Table of Contents

1. Introduction

AvaTax for Communications (AFC) SaaS Pro is easily integrated into your application. Typically, integration efforts are measured in days, instead of the months required of other products. Avalara provides a complete sales and use taxation solution with its AFC SaaS Pro product. AFC SaaS Pro performs transaction processing for numerous types of products and services, from alcohol to prescription drugs, from general merchandise to software sales.

AFC SaaS Pro relieves or minimizes your organization of the following cost of doing business burdens:

1) Cost of research and maintenance of the continually changing tax data of approximately 70,000 communities capable of levying taxes in the United States as well as international taxes.

2) Tax compliance reduces liability and exposure to tax audits.

3) Cost of tracking, maintenance, and implementation of tax systems based upon the whim of federal, state, county, and/or local governments.

4) Cost of research and maintenance of nearly 400,000 tax rates and complex application rules including, but not limited to the following:
   - Over 200 tax types.
   - Federal, state, county, district and local taxes.
   - Taxing for eighty other countries.
   - Applicability to sale and/or resale
   - Applicability to different customer types
   - Taxes that are included in base taxable amount of other taxes
   - Maximum tax bases
   - Excess tax rates
   - Transit Taxes
   - State and county taxes that are replaced by county and /or local taxes
   - Special state and county rates based upon county or local jurisdiction
   - Determination of 1 of 10 different jurisdiction determination rules per tax
   - Maintenance of address to tax jurisdiction cross reference data
   - Application of taxes based upon transaction type
   - Application of taxes based upon service type
   - Application of taxes based upon attribute type

1.1 Getting Started

1.1.1 Integration Requirements

Please reference the links provided below for the most current and detailed integration requirements.

- Integration Checklist:
  https://developer.avalara.com/certification/communications/

- Certification Use Cases:
  https://developer.avalara.com/certification/communications-certification/

1.1.2 Key Input Requirements

The following items are required as input for AFC SaaS Pro:

- Transaction Date (Please reference Transaction Date for additional details).

- Jurisdiction information details: (Please reference Specifying a Tax Jurisdiction for additional details).
  - Address
  - Zip
  - Zip+4
  - FIPS Code
  - PCode
  - NPANXX

- Product Information/ Transaction and Service Types (Please reference AFC Transaction and Service Types for additional details.)

- Sale Type details: (Please reference Transaction for additional details.)
  - Retail
  - Wholesale

- Customer Type Info (of who is being billed) details: (Please reference CustomerType for additional details.)
  - Business
  - Residential
  - Industrial
  - Senior Citizen

- Business Class (of who is billing) details: (Please reference BusinessClass for additional details.)
  - CLEC

- o ILEC
- o Other

- Service Class (of who is billing) details: (Please reference ServiceClass for additional details.)
    - o Primarily Long Distance
    - o Primarily Local
    - o Other

- Other Company Info (of who is billing) details: (Please reference Transaction for additional details.)
    - o Regulated: Y or N
    - o Franchise: Y or N
    - o Facilities: Y or N

*Certain Inputs can be set as defaults and do not have to be selectable options in the billing platform being utilized. This represents a selection of data elements. Additional features/functions may require additional elements.

1.2   AFC Transaction and Service Types

The AFC SaaS Pro software package provides a wide range of transaction and service types. The result is a complete taxation package.   When passing a transaction to AFC SaaS Pro, the user must indicate the transaction and service type of the transaction. See **TM_00505_AFC Telecom Mapping Guidelines.pdf** for more details about the transaction and service types supported by AFC SaaS Pro.

1.3   Client Information

Many taxing jurisdictions apply taxes differently to different types of customers. In addition, many jurisdictions apply taxes differently depending upon the type of sale. For these reasons, it is necessary to indicate to AFC SaaS Pro the type of client the transaction occurred with. The client type will be "Business", "Residential", "Industrial" or "Senior Citizen". Likewise, it is necessary to indicate the type of sale, "sale" or "resale". Transactions that are performed with the end user of that product or service are "sale". Products that are with a reseller (entity that intends to resell the product or service purchased) are "resale". To have exempt taxes available for reporting, exemption type 3 (Sales For Resale) should be used in combination with Resale.

1.4   Transaction Date

AFC SaaS Pro maintains multiple tax rates for every tax contained in the system. One rate is the current tax rate and the others are the previous tax rates. When transactions are passed to AFC SaaS Pro for tax generation, the caller is required to specify a date. AFC SaaS Pro compares this date to the effective date of each tax that applies to the transaction. If the date passed to AFC SaaS Pro is "equal to" or "greater

than" the effective tax date, the current tax rate is used. If the date passed to AFC SaaS Pro is prior to the effective date, AFC SaaS Pro will select the previous tax rate for the tax based on the effective date of the previous tax rate. If a transaction is passed to AFC SaaS Pro without a date (that is, the date is set to zero), AFC SaaS Pro will set the date to the current date.

The Date field is normally populated with the bill date, invoice date or call date (as applicable) and may appear in one of the following Standard Date and Time Format Strings for .NET Framework shown in the table below.

Date Format	Sample
mm/dd/yyyy	06/01/2016
m/d/yyyy	6/1/2016
mm-dd-yyyy	06-01-2016
m-d-yyyy	6-1-2016
yyyy-mm-dd	2016-06-01
yyyy-m-d	2016-6-1
yyyy-mm-ddThh:MM:ss	2016-06-01T13:45:30
yyyy-m-dThh:MM:ss	2016-6-1T13:45:30

A common question asked for some transactions is "What date should be used for taxing a transaction, the date it occurred or the billing date?"   Normally this should be the billing date. Generally accepted accounting principles tell us we should record our liabilities when we record our revenues. In most cases neither of these are recorded (or even known) until billing occurs. Large organizations may have a different answer to this question. If you are running high volumes of transactions daily and record revenue as it occurs, then the tax should be recorded on the same basis (i.e. the transaction date should be used).

Note: The invoice date passed to the server in the transaction by default is preserved as is. It is recommended that clients not use time zone modifiers on the invoice date.

1.5   Tax Types

The application of taxes varies from location to location as well as the particular transaction and service provided. Different localities typically have different taxes and logic associated with the application of the taxes. For example, one locality may apply tax to medical equipment paid for by Medicare but not to equipment paid for by Medicaid. A neighboring locality may apply tax to both or neither. For the most current list of Tax Types, see the **TM_00505_AFC Telecom Mapping Guidelines.pdf** file in the most current Distribution/Update download. This document is under /support/docs.

1.6   Bundles

The Bundling feature allows the user to define multiple AFC SaaS Pro transaction/service type pairs as one unique "bundle" transaction/service pair type. AFC SaaS Pro processes all of the transaction/service type pairs in the bundle and stores the results in the file EZtax.bdl, the formatted text bundle definition file. This file must be created and edited by AFC Bundler, an optional utility program. The bundle (.bdl) file must be installed on our web server to be implemented. Email your zipped bundle file to communicationsupport@avalara.com for installation. We will contact you when it has been installed.

1.7   Overrides

Overrides allow the client to change the rate of a tax in the AFC Engine. Avalara markets the AFC Override Utility (a Graphic User Interface based Windows program that is sold separately) to support this activity. It steps the user through the process of creating an EZtax.ovr file.

<div align="center">

WARNING

An override to exempt taxes OVERRIDES the tax information in Avalara's tax research database. This is not recommended for those that do not possess a full understanding of the tax ramifications and liabilities when doing so.

</div>

The override (.ovr) file must be installed on our web server to be implemented. Email your zipped override file to communicationsupport@avalara.com for installation. We will contact you when it has been installed.

Overrides on tax rates may also be performed for specific transactions.

1.8   Exclusions

The AFC SaaS Pro Telecom interface allows the client to specify the states where the client's company wants to exclude all taxes. There are two methods in which exclusions can be specified to AFC SaaS Pro:

(1) Users can provide an exclusion file to be installed on our web service. The format of the exclusion file is either a state abbreviation or the country ISO code followed by a comma and the state abbreviation, one per line. Example: "AZ" or "USA,AZ" are both valid for excluding taxes in the state of Arizona. The exclusion file should be emailed to communicationsupport@avalara.com for installation. We will contact you when it has been installed.

(2) Users can provide an array of Exclusion objects with each tax method call to the telecom interface. These exclusion objects modify the exclusion settings only for the current method call.

If both methods are utilized, the exclusions passed in via the API call will be used and the exclusion file will be ignored for that particular transaction.

Note: Excluded tax jurisdictions will either appear as unknown or will not be included in any Transaction Service Reports (TSRs) produced.

1.9 Exemptions

There are three types of exemptions allowed in AFC SaaS Pro.

1.9.1 Category Exemptions

Category Exemptions are exemptions applied by tax category. (Please reference the table below to view the current listing of tax categories. Users may also reference **Section 6 Category Definitions** in the **AFC Telecom Mapping Guidelines** document for a detailed overview of tax type assignment per tax category.)

Tax Categories	
Category ID	Name
0	No Category Description
1	Sales and Use Taxes
2	Business Taxes
3	Gross Receipts Taxes
4	Excise Taxes
5	Connectivity Charges
6	Regulatory Charges
7	E-911 Charges
8	Utility User Taxes
9	Right of Way Fees
10	Communications Services Tax
11	Cable Regulatory Fees
12	Reserved
13	Value Added Taxes

The Category ID number must be passed in order for exemptions to be applied appropriately. In addition to this, the Country Code or State Abbreviation or both values for the category exemption must be provided as well. Please reference the example provided using the following scenario.

Category	Country	State	Comments
1 – 13	USA		Provided categories will be exempted in all US jurisdictions.
	USA		Error message will be generated as Category is required field.
1 – 13	USA	KS	Provided categories will be exempted in all Kansas jurisdictions.
1 – 13		KS	Provided categories will be exempted in all Kansas jurisdictions by assuming "USA" as country.

1.9.2 Level Exemptions

The exemption level is the jurisdictional level of the taxing authority that defines the tax. It is used to exempt taxes at specific federal, state, county and/or local level taxes.

Note: Most Federal taxes are only exempted when selling to a reseller who is registered, reporting, and remitting to the regulating agency. For this reason, a wholesale exemption or a tax type exemption must be used to exempt taxes at the Federal level.

1.9.3 Tax Type Exemptions

Tax Type exemptions are used to specify a specific Tax Type at a specific Tax Level to be exempted for the current transaction. The exemption jurisdiction code specifies the jurisdiction for the tax exemption. If the jurisdiction code is not specified (i.e. set to zero), then all taxes of the Tax Type and Tax Level specified are considered exempt regardless of the jurisdiction they are calculated for. Typically the PCode should be specified as tax type exemptions are normally only effective for specific jurisdictions.

Another option allows the tax type to be set to zero, to indicate that all taxes of a specific tax level are exempt in the specific jurisdiction.

Note: To exempt a county or city for all taxes, apply a specified exemption for the county and/or city by entering the tax type as 0. This is the wildcard character for tax type when applying exemptions in AFC SaaS Pro.

1.10 Compliance Files

The web service automatically generate compliance files at the end of the month for the clients contracted for compliance services. These files contain a summary of the tax data generated for all transactions processed throughout the month. The format of the compliance files may be that of any of the available AFC SaaS Pro sorting and reporting utilities. The compliance files may be placed in an FTP site where they can be downloaded or they may be automatically emailed to a specific email account. AFC SaaS Pro Clients are required to contact communicationsupport@avalara.com in order to set up these preferences.

<u>**Sorting & Reporting Utilities**</u>

Method	Description	Example
Date Method 0	This method gathers all transactions that were sent to our server during the previous month by looking at start time field for each transaction and generate the compliance files from that data on the first of every month or requested date. AFC SaaS Pro uses date method 0 to generate compliance reports, unless specified by client.	• **Reporting Scenario 1:** Throughout the month of January 2013, you submit several transactions to the web service for taxation at the time each sale occurs. The system will generate the January 2013 compliance reports on February 1st, 2013 for all the transactions received in January, irrespective of invoice date.

Method	Description	Example
Date Method 1	This method processes data based on invoice date. We gather all transactions that have an invoice date of the previous month and generate the compliance files from that data on the first day of every month.	• **Reporting Scenario 2:** Throughout the month of January 2013, you submit several transactions to the web service for taxation with an invoice date of January 2013. On February 1st, 2013, the system will gather all transactions with an invoice date of January 1st through January 31st 2013 and generate the compliance data. • **Reporting Scenario 3:** You send a transaction with an invoice date of January 15th past February 1st when AFC SaaS Pro runs your compliance reports, this transaction will never show up on any report. You will need to submit a request to our network services to rerun your data for January in order to capture January data sent past February. Additional charges apply to rerun, or provide subsequent compliance reports for the month. Alternatively, you may request your compliance reports to be generated on a specific day of the month in order to give you enough time to complete processing tax calculations for your billing cycle. Please contact communicationsupport@avalara.com in order to set up these preferences.

Method	Description	Example
Date Method 2	This method processes data based on an optional10 field. The client will fill in the optional10 field with the year and month (YYYYMM format) that they want the transaction to show up in their compliance files. When AFC SaaS Pro runs the compliance reports, it only includes transactions from the previous month, based upon optional10 field.	• **Reporting Scenario 4:** Throughout the month of January 2013, you submit several transactions to the web service for taxation at the time each sale occurs. Each transaction must include the value 201301 in the Optional10 field. The system will generate the January 2013 compliance reports on February 1st, 2013 and include the taxes for all transactions with a 201301 value in the Optional10 field. It is important to note that if you send transactions with 201301 in the optional10 field, after the compliance report is run on February 1, they will be provided in any compliance reports. In this case, you should record the transaction as 201302 or a future period for them to be reported upon. • **Reporting Scenario 5:** On February 1st 2013, you process a batch for all January 2013 transactions. Each transaction must have a 201301 value in the Optional10 field. You should request your reports to be generated on the 2nd day of the month beforehand. The system will generate your January 2013 compliance reports on the 2nd day of the month and include the taxes for all transactions with a 201301 value in the Optional10 field. • **Reporting Scenario 6:** On February 15th, you process an adjustment for a transaction that took place on January 15th 2013. The Date field of the transaction must have the date of the original transaction (January 15th) in order to get the appropriate tax adjustments back, but the Optional10 field of the transaction must have a value of 201302 since the adjustment was handled during February. The tax adjustment will be included in your February 2013 compliance reports.

In order to allow enough time to complete tax calculations for your billing cycle, Network Services will assist in establishing the correct date method depending on your transaction method and the appropriate date of the month for your compliance reports. Please contact communicationsupport@avalara.com in order to set up or establish the appropriate preferences.

1.11 Live vs. Test Transactions

Each client is provided with a three-character company code when signing up for the AFC SaaS Pro product. In order to run a live transaction, the client's company code must be placed in the CompanyIdentifier field of the Transaction to be processed using AFC SaaS Pro (see Transaction data definition). Only taxes for transactions that contain this code in the CompanyIdentifier field will be included in the monthly compliance files created by the AFC SaaS Pro Web Service. The

CompanyIdentifier field may be left blank or use a value other than the assigned company code in order to process transactions where the taxes generated are to be excluded from the monthly compliance reports such as test transactions or quotes.

1.12   Compliance Month

Optional10 field applies when Date Method 2 is used. The month for the billing cycle must be specified in the Optional10 field of the transaction by entering the year and month in YYYYMM format. This will allow the transaction and taxes to be reported in the appropriate compliance reports for the specified month.

1.13   Tax Adjustments

Tax adjustment functions are used for adjustment activities such as refunds, changing a customer's bill or writing off un-collectable accounts. Please apply the following general rules while passing tax adjustments via Web Service.

1. To calculate adjustments accurately:
    a. Provide the correct web adjustment method (Please refer to Telecom Web Methods for more details).
    b. Send positive values for charge, line, and locations.
    c. Send the adjustment method (see Adjustment Method Table).
    d. Send the discount type (see Discount Type Table).

1.14   Interstate/Intrastate Determination

The Interstate or Intrastate feature provides clients with the ability to apply interstate or intrastate charges appropriately when sending transaction messages to the AFC SaaS Pro service without specifying either the transaction type or the service type. To use this functionality, set the transaction type to '-1' and a valid service type listed in the below scenarios; conversely, set the transaction type to a valid transaction type listed in the below scenarios and the service type to '-1'. The AFC SaaS Pro service will determine the appropriate interstate or intrastate transaction or service and return the appropriate taxes.

This feature is enabled for the following APIs:
CalcTaxesWithZipAddress
CalcTaxesWithFipsCode
CalcTaxesWithPCode
CalcTaxesWithNpaNxx
CalcProRatedTaxes
CalcAdjWithZipAddress
CalcAdjWithFipsCode
CalcAdjWithPCode
CalcAdjWithNpaNxx

CalcProRatedAdj
CalcTaxesInCustMode
CalcTaxesInCustModeV2
ProcessCustomerBatch
ProcessCustomerBatchV2

Note: This feature is only supported with use of the APIs listed above.

In addition, this feature is only applicable in each of the following scenarios:

A) **Scenario A** – One of the following service types listed in the table below has been identified, but the transaction type needs to be determined. For example, a service type of '1' is entered for a toll service and '-1' is entered for the unknown transaction type. The web service will determine the correct transaction type and apply INTERSTATE (transaction type 1) or INTRASTATE (transaction type 2) based on the termination and origination data.

Service Type	Service Type Description
1	TOLL
2	TOLL FREE
3	WATS
4	PRIVATE LINE
14	LATE CHARGE
16	900
27	DATA
54	DIRECTORY ASSISTANCE
635	TOLL FREE NUMBER

B) **Scenario B** – The following transaction type has been identified, but the service type is unknown. For example, '61' is entered for a VPN transaction and '-1' is entered for the unknown service type. The web service will determine if the service type is INTERSTATE MPLS (service type 585) or INTRASTATE MPLS (service type 586) based on the termination and origination data.

Transaction Type	Transaction Type Description
61	VPN

C) **Scenario C** – One of the following transaction types listed in the table below has been identified, but the service type is unknown. For example, '19' is entered for a VOIP transaction and '-1' is entered for the unknown service type. The web service will determine if the service type is INTERSTATE USAGE (service type 49) or INTRASTATE USAGE (service type 50) base d on the termination and origination data.

Transaction Type	Transaction Type Description
19	VOIP
20	VOIPA

Transaction Type	Transaction Type Description
21	PAYPHONE
59	VOIP NOMADIC

1.15 Private Line

The Private Line feature provides clients with the ability to obtain taxes for a private line transaction. Clients must call with one of the following APIs:

 CalcTaxesWithPCode
CalcTaxesWithFipsCode
CalcTaxesWithNpaNxx
CalcTaxesWithZipAddress
CalcAdjWithPCode
CalcAdjWithFipsCode
CalcAdjWithNpaNxx
CalcAdjWithZipAddress
CalcTaxesInCustMode
CalcTaxesInCustModeV2
ProcessCustomerBatch
ProcessCustomerBatchV2

In addition, clients must specify that the transaction is for a private line and provide a number between 0 and 1 to indicate the percentage which applies to the origination point. Any remaining charges are then applied to the termination point.

Note: This feature is only supported with use of the eight APIs referenced. It is not supported for use with Tax Inclusive or Reverse APIs.

1.16 Proration

The Proration feature provides clients with the ability to calculate prorated taxes on a transaction that represents a partial month of service through use of the CalcProRatedTaxes API or the CalcProRatedAdj API. Percentage-based taxes are prorated by way of the charge amount passed. However, fixed and per line taxes are subject to special pro-rating rules and procedures. Some tax authorities allow sellers to pro-rate fixed and per line taxes for partial months, but many insist on receiving the full amount. AFC SaaS Pro will apply these rules automatically if the pro-rating feature is used. . A number between 0 and 1 is passed to specify the percentage of the month the service was active. The pro-rating functionality checks the logic of each tax in the appropriate jurisdiction to determine if prorating is allowed. If prorating is not allowed, the full amount is taxed. If pro-rating is allowed, the fixed or per line tax applicable to the service will be returned multiplied by the fraction supplied.

If the proration is being used for an adjustment credit rather than a partial charge, the ratio of the percentage applied should reflect the portion of the month in which the service was not active.

1.17 Tax Grouping

The following options determine how tax calculation results are returned after calling one of the tax calculation API functions. By changing this option, the taxes returned by the tax calculation API function may be grouped according to this setting.

NOTE: This option will not modify the way that tax calculation results are logged into the AFC log data. Only the tax calculation results returned by either one of the tax calculation API will be group according to the values set for this option.

Please contact communicationsupport@avalara.com in order to have tax grouping options set or activated.

General Rules
The following rules apply when using any value for this option:

1. Federal taxes may not be grouped. Each Federal tax will be returned individually.
2. Non-billable taxes may not be grouped.
3. Only rate-based taxes may be grouped. Taxes with a different calculation type (for example, fixed, per line, etc) will be returned individually.
4. Use taxes may be grouped with other use taxes only (for example, state and local use tax). Use taxes will not be grouped with other tax types.
5. When grouping taxes for different tax levels (for example, state and local taxes) the jurisdiction code for the lowest level jurisdiction will be returned.
6. Unincorporated taxes will be considered as County taxes when grouping taxes by tax level, and will be grouped accordingly.
7. The tax rates for all taxes being grouped into a single record will be added together.

Default Option:
- **groupresults=default (default)**
  The default option indicates that the taxes returned in the tax table after processing a transaction with AFC will not be grouped. Each tax will be returned in an individual tax record.

Tax Level Options:
- **groupsamelevel**
  This option will cause rate-based taxes at the same level to be grouped together. For example, if AFC returns a Local Sales Tax and a Local District Tax, these taxes will be grouped together since they have the same tax level (Local).
- **groupstate_groupcountyandlocal**
  This option will cause all state taxes to be grouped together into a single record, and all county and local taxes to be grouped together into a separate record.

- **groupstatecountyandlocal**

This option will cause all state, county, and local taxes to be grouped together into a single record.

Sales Tax Options:
The following options may be used in combination with any of the tax level options specified above. These options may be used to group sales taxes separately from other taxes. This option must be appended to the EZTax.cfg file following the line for the tax level group option.

- **groupsales**
  This option will group Sales Taxes (tax type 1) and Use Taxes (tax type 49) taxes into a separate record according to the tax level option being specified.

- **groupsalescategory**
  When using this option, AFC will group any items that are considered Sales Taxes together. In addition to the Sales Tax (tax type 1) and Use Tax (tax type 49), some District and Transit taxes are also in the sales tax category.

Examples
- **groupstatecountylocal**
  By entering the configuration option shown above into the EZTax.cfg file, all State, County, and Local taxes will be grouped together into a single record.

- **groupstatecountylocal**
- **groupsalescategory**

By entering these two options in the configuration file, all State, County, and Local sales category taxes will be grouped together into a single record. Any other State, County and Local taxes (if any) will be grouped into a separate record.

Tax Return Table
When grouping taxes together, the fields in the tax return table will contain the following values:

1. Jurisdiction Code. Jurisdiction Code (PCode) for the lowest level jurisdiction. For example, if Kansas state taxes and Overland Park local taxes were grouped together, the tax record will contain the jurisdiction code for Overland Park.
2. Tax level. When grouping State, County and Local taxes together, the tax record will contain a value of 6 in the tax level. When grouping only County and Local taxes together, the tax record will contain a value of 7 in the tax record. Constants are provided for these values in the appropriate file.
3. Tax type. When grouping different taxes together, the tax type in the tax record will contain a value of 0. If only Sales Taxes (tax type 1) or Use Taxes (tax type 49) are being grouped together, the tax record will contain the corresponding tax type.
4. Tax amount. This field will contain the sum of the tax amount for all taxes being grouped together.
5. Tax rate. This field will contain the sum of the tax rates for all taxes being grouped together.

NOTE: *The remaining fields in the tax table will not contain any meaningful value. Grouping tax calculation results may serve as a way to simplify the tax information for display purposes only. If further detail is required for each tax being returned by AFC, this feature should not be used.*

1.18 Specifying a Unique Identifier

AFC SaaS Pro does not provide in the response a unique identifier for a specific transaction. However, there are numerous reporting fields which can be used for the requesting application to populate a unique identifier. Reporting fields do not impact tax calculations. It is recommended that clients generate or construct the unique identifier in a way that is meaningful to their billing application and then proceed to provide that data to Avalara through one of the available reporting fields.

Reporting fields that can be used individually or in combination to represent a unique identifier are shown in the table below.

Reporting Fields	
Column Name	Description
Optional Optional4 Optional5 Optional6 Optional7 Optional8 Optional9 Optional10 InvoiceNumber ServiceLevelNumber	Unsigned int. Should be >= 0. The max value is different for different platforms, but in general supports values at least up to 4294967295 (2^{32} -1).
OptionalAlpha1 CustomerNumber	Alpha-numeric field. Supports up to 20 bytes.
OptionalFields	Alpha-numeric field. Supports up to 10 fields and up to 150 bytes per field.

1.19 Commit/Uncommit

The process to commit documents is optional and used to identify documents that should be included in the Compliance Reports for remittance to the Department of Revenue (DoR) for tax compliance reporting. This process is often used when not all taxes being calculated are considered final for compliance reporting. When a document is considered final, a commit can be sent to finalize the document. This also allows users to commit an entire invoice by using the document code instead of calculating the taxes again when the document is determined to be final.

Note: This feature is supported with the use of all tax calculations. Please reference the **AvaTax for Communications SaaS Pro Commit/Uncommit Process** document for additional information and details.

1.20 Application of Tax Brackets and Limits

1.20.1 Tax Brackets

Some jurisdictions will dictate a tax rate that changes as the taxable amount of the transaction increases. These break points at which the changes occur define the brackets (or steps) and are most commonly based on dollar amount ranges although other units of measure exist. The rate may increase or decrease according to usage levels.

AFC SaaS Pro supports these transactions with an unlimited number of tax brackets. The Avalara Tax Research department continually researches jurisdictions for specific tax practices, such as tax rate brackets, updating the AFC Engine monthly. These updates occur automatically and the user is not required to make changes to account for this.

As an example of applying tax brackets, if a jurisdiction has a general sales tax set at 2% for the first $500 of a single transaction and set at 1% for that which is over $500, the tax for a $1200 sale would result in ($500 x 2%) = $10 plus ($700 x 1%) = $7 which is a total tax of $17.00.

1.20.2 Tax Limits

Some jurisdictions have established tax rates that either take effect or cease to take effect at a specific threshold, defined as a currency value. The point at which this occurs is referred to as a cap or limit. AFC SaaS Pro supports these transactions and the user is not required to make changes to account for it.

As an example of applying tax limits, if a jurisdiction charges a 10% UTT on only the first $10 of an invoice, the tax for a $20 invoice would "cap" at the $10 threshold, resulting in a ($10 x 10% =) $1 UTT fee.

As an example of the converse, if a jurisdiction does NOT tax the first $25 of Internet Access usage, a $35 charge would be reduced by the $25 threshold "limit," resulting in a ($35-$25 =) $10 taxed amount.

1.21 Use of Client IDs in the REST Interface

Users may obtain access to multiple client accounts which are referred to as Client IDs when using AFC SaaS Pro. The Client ID is assigned and provided by Avalara. They are specific to each client account; however, an individual user may utilize several Client IDs based on the number of client accounts to which they have been assigned to manage.

Please reference the image below for further illustration of the relationship between individual users and Client IDs.
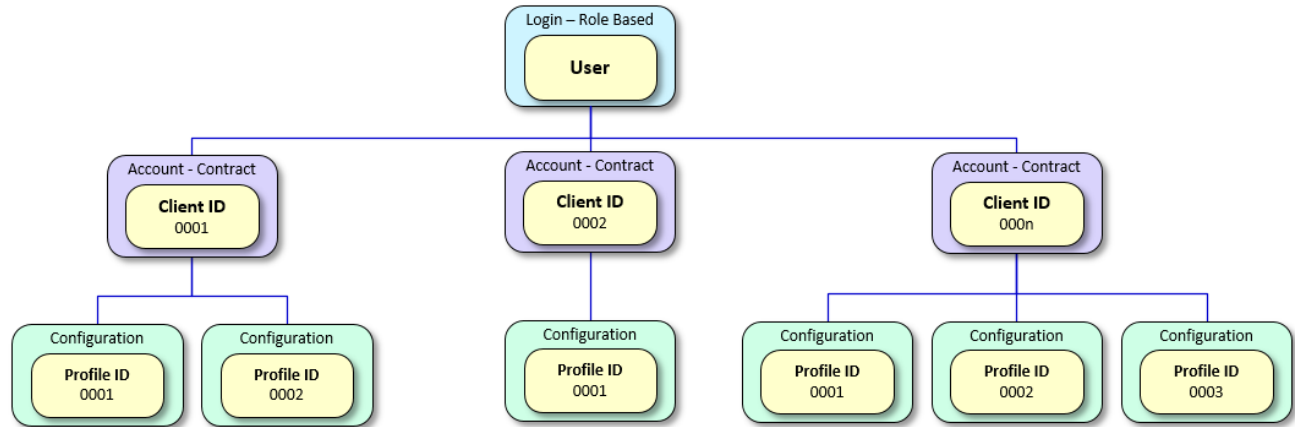
1.21.1 Definition of Terms

Term	Definition
User	The username or email account login used to access the AFC Customer Portal. This includes Reporting, Developer Content and Tax Determination access. The user account will have one to many AFC client accounts associated with it. Please contact communicationsupport@avalara.com to create or change a user account.
Client	An AFC entity with a valid contract. The client account will have one to many profiles. Please contact your Avalara Customer Account Manager (CAM) to establish a new client.
Profile	The client's customizations for the AFC tax engine. This includes overrides, bundles and exclusions.

2. Specifying a Tax Jurisdiction

For AFC SaaS Pro to calculate taxes for transactions correctly, it must first determine the taxing jurisdiction for the transaction in question. The tax laws of various jurisdictions complicate this. AFC SaaS Pro currently supports multiple unique rules for determination of the correct tax jurisdiction. Jurisdiction determination is usually based upon three inputs to AFC SaaS Pro:

1. The Service Address or the Bill To Number (BTN).
2. The termination location of the transaction (for telephone calls only) to be taxed. The number called, also known as the "To Number" or the "Termination number" usually specifies this.
3. The origination location of the transaction to be taxed. For telephone calls this is usually specified by the number called from, also known as the "From Number" or Origination number.

The jurisdiction, for the majority of telecommunications taxes applied by AFC SaaS Pro, is determined by the Goldberg or "2 out of 3" rule. With this particular rule, the three jurisdiction inputs pointed out above

are compared. If 2 of the 3 jurisdictions supplied match, that is the jurisdictions for which taxes are generated. It is slightly more complicated since there are four authority levels for which jurisdictions determination must be made (i.e. federal, state, county, and local), however; this brief discussion illustrates the concept. Remember, the Goldberg Rule although the most common, is only one of 10 rules specified for jurisdiction determination by taxing jurisdictions in the United States.

Jurisdiction information can be supplied to the system in different ways. It can be supplied using a permanent jurisdiction code (PCode), using a FIPS Code, using an NPANXX, or using zip code and address information. AFC SaaS Pro allows the user to identify the jurisdictions by any of these methods, thereby providing maximum flexibility. AFC SaaS Pro allows the specification of jurisdiction information using any combination of these methods during the same session.

PCode

PCodes are permanent jurisdiction codes that Avalara provides that allow AFC SaaS Pro software users to populate their databases with jurisdiction information. With PCodes, clients can populate their customer records with jurisdiction information and never worry about changes of jurisdiction codes. If a jurisdiction code changes, Avalara re-maps the PCodes so clients are not affected.   This allows Avalara's clients to populate client records with a PCode.

NPANXX

When using NPANXXs to specify the jurisdiction, some special considerations must be addressed by the user. The first issue concerns toll free 800 and 888 numbers.   The first 6 digits of these numbers do not constitute an NPANXX. Each 800/888 number is associated with a "ring to" number. The "ring to" number is the number that is actually reached when the 800/888 number is dialed. The NPANXX of the "ring to" number should be used as the NPANXX for the 800 number. This replacement must be made before the information is passed to the AFC SaaS Pro system. These issues also arise with 900 numbers.

Another item that requires special attention is account codes. Many switches are capable of producing CDRs with account codes in place of a "Bill To" number. Once again, this number is not related to an actual NPANXX. The user is required to convert this number to the actual NPANXX using data from the billing system before interfacing with AFC SaaS Pro.

NPANXXs exist for the following countries U.S., Barbados, Canada, Guam, Mexico, Northern Mariana Islands, Puerto Rico, Trinidad, Tobago and U.S Virgin Islands. These countries are on the North American dialing plan.

FIPS Code

FIPS Codes are issued by the National Institute of Standards and Technology (NIST). AFC SaaS Pro provides internal translation tables from FIPS Codes to PCodes, so that using FIPS Codes is almost as fast and accurate as using PCodes. Some special taxing districts are not identified in separate FIPS codes, so there is some loss of accuracy, but the majority of transactions will produce the same tax results with FIP Codes as with PCodes.

ZIP Code

The accuracy of the ZIP code method depends upon the amount of data provided for the address as well as the user's ability to choose the correct taxing jurisdiction zip code and address. AFC SaaS Pro databases contain numerous duplicate zip codes that cross not only taxing jurisdiction boundaries, but boundaries of localities as well. Providing a complete address along with the zip code insures the best match possible. When address information is missing, AFC SaaS Pro returns taxes based upon the first match of the provided input information.

It is appropriate to use the zip code interface or PCode methods for transactions where the jurisdiction can be positively identified by the calling application.

Note: For the United States, clients may enter various forms of the zip code plus 4 in the ZipCode and ZipP4 fields (with or without a dash or space). Please reference the table provided below for examples of the alternate formats which can be entered for US zip codes.

United States	
ZipCode Field	**ZipP4 Field**
12345	1234
12345 1234	
12345-1234	
123451234	

For Canada, clients may populate the 6-digit Postal Code in the ZipCode field (with or without a dash or space). Canadian Postal Codes may also be entered by populating the first three in the ZipCode field and the last three in the Zip plus 4 (ZipP4) field.

For example, the Canadian zip code A1A 0A0 may be entered in any of the formats featured in the table below.

Canada	
ZipCode Field	**ZipP4 Field**
A1A 0A0	
A1A-0A0	
A1A0A0	
A1A	040

<u>Tax Jurisdiction</u>

AFC SaaS Pro provides the ability to obtain the PCode of the tax jurisdiction for a transaction by passing jurisdiction and transaction/service pair information.

NOTE: The transaction/service pair is required and must be provided in order to determine the correct tax jurisdiction.

2.1   Getting the Right Tax Jurisdiction for Local Taxation

It is important to get the end users location correct for local taxation. People tend to know what they are charged for local taxes. These taxes also have a tendency to change from one neighborhood to the next.

Avalara provides a comma delimited ASCII file to aid with entry of customers in your system and to help insure the correct taxing jurisdiction is setup. The file is "all_adr.txt". This comma delimited ASCII files is provided to allow AFC SaaS Pro clients to populate database tables in their system with this information. The all address (all_adr.txt) file is a cross-reference of locations to PCodes. The format of the files is illustrated below.

The "all_adr.txt" file format and example follows:

```
Pcode,P/A,Country,State,County,City,"Zip code range start","Zip code
range end"
.
3346800,0,USA,PA,LANCASTER,BART,17503,17503
3346900,0,USA,PA,LANCASTER,BAUSMAN,17504,17504
3347000,0,USA,PA,LANCASTER,BIRD IN HAND,17505,17505
3347100,0,USA,PA,LANCASTER,BOWMANSVILLE,17507,17507
3347200,0,USA,PA,LANCASTER,CHRISTIANA,17509,17509
3347300,0,USA,PA,LANCASTER,CHURCHTOWN,17555,17555
3347400,0,USA,PA,LANCASTER,CONESTOGA,17516,17516
3347500,0,USA,PA,LANCASTER,DRUMORE,17518,17518
3347600,0,USA,PA,LANCASTER,EAST EARL,17519,17519
3347700,0,USA,PA,LANCASTER,ELM,17521,17521
3347800,0,USA,PA,LANCASTER,GAP,17527,17527
```

3. Invoice Mode

3.1 Overview

AFC SaaS Pro applies Brackets and/or Limits on a per transaction basis unless operating in Invoice Mode. Invoice Mode is used to group transactions that apply to the same customer. When in Invoice Mode, AFC SaaS Pro maintains a history of the transactions and applies the Brackets and/or Limits to the entire group of transactions.

NOTE: Please reference Application of Tax Brackets and Limits for additional details, descriptions and examples of tax brackets and limits.

AFC SaaS Pro provides the ability to process up to 30,000 transactions within an invoice for a single transaction or up to 50,000 transactions within an invoice for a batch transaction in Invoice Mode. Tax calculation results will be summarized by jurisdiction and tax type. Optionally, the individual taxes for each line item can also be returned in the output, but be advised that the response size may be up to a couple megabytes and may take up to one minute to process depending on the number of transactions in the customer batch. In addition, it is recommended to increase the timeout of your web API calls to 10 minutes although response times are expected to be much shorter even or the largest batches.

NOTE: The Invoice Mode feature must be individually activated for each client. Please contact communicationsupport@avalara.com in order to access this functionality.

Also, it is recommended that transactions in Invoice Mode are contained within one monthly billing cycle.

3.2 Batch Transaction Submission

The CalcTaxesInCustMode APIs can be used for submitting a batch of telecom and/or sales and use transactions and adjustments within a single SOAP request in order to be processed using Invoice Mode. The steps for processing an invoice in using this method are the following:

1. Invoke the CalcTaxesInCustMode API to calculate taxes on all transactions that have been submitted. This API accepts a parameter indicating if the detailed taxes for each line item should be returned or if only the summarized taxes should be returned.

3.3 Single Transaction Submission

The following APIs can be used in order to submit each transaction within an individual SOAP request. The advantage of using this method is that each transaction can be validated prior to processing the entire batch. The steps for processing an invoice in using this method are the following:

1. Invoke the BeginCustomerBatch API to flag the beginning of a new customer transaction batch. AFC SaaS Pro will return a customer batch ID which will be used when submitting each transaction to the web service.

2. Submit each line item within the invoice by invoking one of the Invoice Mode APIs (CalcCustTaxes, CalcCustAdj, SAUCalcCustTaxes, or SAUCalcCustAdj) and pass the customer batch ID along with the telecom or sales and use transaction as parameters to the API call. The web service will not calculate taxes on the transaction at this point. It will simply keep track of each transaction in the batch.

3. Invoke the ProcessCustomerBatch API to calculate taxes on all transactions that have been submitted for that customer batch ID. This API accepts a parameter indicating if the detailed taxes for each line item should be returned or if only the summarized taxes should be returned.

After invoking ProcessCustomerBatch, the customer batch ID will no longer be valid.

Note: It is recommended that the batch transaction method be attempted first and that the single transaction method only be utilized if the batch transaction method does not work.

3.4 Taxing Jurisdiction Specification

Invoice Mode APIs accept either combination of PCodes, FIPS Codes, Zip Address, or NPANXX for the bill-to, origination, and termination. However, make sure that only one property is assigned for each location. For example, if setting a value in the BillToPCode field, leave the BillToFipsCode, BillToAddress, and BillToNpaNxx fields null.

3.5 API Results

The response from any of these batch mode APIs contain two fields:

- SummarizedTaxes. This is an array of CustomerTaxData objects containing the accumulated taxes grouped by tax type and jurisdiction.
- Taxes. This is an array of TaxData objects containing the individual tax calculation results for each transaction in the batch. This array is populated only if the returnDetail parameter for the Invoice Mode API parameter is set to true. It is recommended to place a unique identifier within one of the optional fields of the transaction in order to be able to match the taxes in this array to its corresponding line item. (Please see Specifying a Unique Identifier for additional details).

3.6   Supported and Non-Supported Features

Not all options for all features are supported in Invoice Mode. Invoice Mode only supports tax calculation methods. Please reference the table below for a high-level overview of supported features within this functionality.

Invoice Mode Supported Features
Standard Tax Calculation Features Supported
Tax calculations with FipsCode
Tax calculations with NpaNxx
Tax calculations with PCode
Tax calculations with ZipAddress
Additional Features Supported
Adjustments
Exclusions
Exemptions
Extended Optional Fields
Private Line*
Safe Harbor Overrides for Traffic Studies
Tax Inclusive**

Support for the Private Line feature DOES NOT include adjustments.

**The Tax Inclusive feature is supported only through use of the Tax Inclusive flag. It is not supported through use of the API.*
.

Invoice Mode Non-Supported Features
Additional Features Not Supported
Bridge Conferencing
Interstate/Intrastate Determination
Overrides via APIs
Proration
Tax Inclusive APIs

3.7   Tax Inclusive Transactions

In order to process tax inclusive tax calculations in Invoice Mode, set the TaxInclusive property of the Transaction field to true. AFC SaaS Pro will determine the appropriate base sale amount required to arrive at the total desired charge. The calculated base sale amount will be included in the TransCharge field of the corresponding TaxDataV2 objects returned for that transaction.

3.7.1   Tax Inclusive - Batch Transaction Submission

The CalcTaxesInCustModeV2 APIs can be used for submitting a batch of telecom and/or sales and use transactions and adjustments within a single SOAP request in order to be processed using Invoice Mode. The steps for processing an invoice in using this method are the following:

1. Invoke the CalcTaxesInCustModeV2 API to calculate taxes on all transactions that have been submitted. This API accepts a parameter indicating if the detailed taxes for each line item should be returned or if only the summarized taxes should be returned.

3.7.2   Tax Inclusive - Single Transaction Submission

The following APIs can be used in order to submit each transaction within an individual SOAP request. The advantage of using this method is that each transaction can be validated prior to processing the entire batch. The steps for processing an invoice in using this method are the following:

1. Invoke the BeginCustomerBatch API to flag the beginning of a new customer transaction batch. AFC SaaS Pro will return a customer batch ID which will be used when submitting each transaction to the web service.

2. Submit each line item within the invoice by invoking one of the Invoice Mode APIs (CalcCustTaxes, CalcCustAdj, SAUCalcCustTaxes, or SAUCalcCustAdj) and pass the customer batch ID along with the telecom or sales and use transaction as parameters to the API call. The web service will not calculate taxes on the transaction at this point. It will simply keep track of each transaction in the batch.

3. Invoke the ProcessCustomerBatchV2 API to calculate taxes on all transactions that have been submitted for that customer batch ID. This API accepts a parameter indicating if the detailed taxes for each line item should be returned or if only the summarized taxes should be returned.

After invoking ProcessCustomerBatchV2, the customer batch ID will no longer be valid.

Note: It is recommended that the batch transaction method be attempted first and that the single transaction method only be utilized if the batch transaction method does not work.

4. Generating Custom Reports

4.1   Report Process Overview

AFC SaaS Pro provides the ability to dynamically generate a custom report containing the transactional data available in the database for a specified timeframe. The general process is as follows:

1. Invoke the CreateReport API in the web service. The ReportOptions parameter specifies which data to include in the report.
2. The report is processed in the background. The resulting output file is placed in the designated client FTP folder where it can be downloaded. This process takes anywhere from 30-90 minutes to complete.

3. Once the report has been generated and is ready for download, an email notification is sent to the email address(s) included in the ReportOptions.

Note: Prior to the first time executing the CreateReport API call, please notify communicationsupport@avalara.com. If not already established, an FTP account needs to be created and the FTP credentials provided.

4.2   Setting up ReportOptions

4.2.1   BaseReport

BaseReport specifies a report type to be used as a starting template for the customlog report. Additional columns can then be included or sorted upon using CustomLogField. This field only needs to be set when the ReportType field is set to "customlog". For further information, see 4.3 – Report Types and 4.4 - Custom Log Report Columns.

The default value for this field is NULL.

4.2.2   CreateNbaFile

If set to TRUE, all non-billable amounts returned as part of the report request are placed in a separate .nba file. If set to FALSE, the non-billable rows are returned as part of the requested report and no additional .nba file is created. The filename and file extension for the NBA file are specified in the FileName and FileExtension fields.

The default value for this field is 0.

4.2.3   CreateNcaFile

If set to TRUE, all non-compliance amounts returned as part of the report request are placed in a separate .nca file. If set to FALSE, the non-compliance rows are returned as part of the requested report and no additional .nca file is created. The filename and file extension for the NCA file are specified in the FileName and FileExtension fields.

The default value for this field is 0.

4.2.4   CustomLogFields

CustomLogFields is a list of CustomLogField. Specify which column should be used using the Column property, if the Column should be included on the report using the Include property, and if the Column should be used for sorting using the Sort property. For further information about the columns, refer to 4.4 - Custom Log Report Columns.

Setting Include to TRUE includes the column in the output while setting it to FALSE removes the specified column from the output. Setting Sort to TRUE uses the specified column as part of the sort on the report while setting it to FALSE does not use the column in the sort.

For example:

- Setting both Include and Sort to TRUE returns the specified column as part of the output as well as sorts the report using this column.
- Setting Include to FALSE and Sort to TRUE does not include the column in the output, but will sort the report based on this column.
- Setting Include to TRUE and Sort to FALSE includes the column in the output, but does not sort the report based on this column.

Note: The sorting of a custom report is based upon the order in which the columns are passed to the API.

Note: Only transactions and taxes that include the 3-character company code for your account in the CompanyIdentifier field of the input TelecomTransaction or SalesUseTransaction will be included in the report.

4.2.5   EmailAddress

Multiple email addresses may be included and should be separated by a semicolon and space (;). When the report is available on the client FTP site, an email is sent to those addresses contained in the EmailAddress field notifying the recipient(s) that the file is ready for download.

This field is required.

4.2.6   FileExtension

The FileExtension field specifies the file extension for the output file. The period for the extension is not required. For example, "csv".

The default value for this field is csv.

4.2.7   FileName

The FileName field of the ReportOptions parameter specifies the name of the output file.

This field allows a date-time format using the following identifiers:

Identifier	Description
yy	Last two digits of year
yyyy	4-digit year
MM	Month
dd	Day of month
hh	Hour
mm	Minutes
ss	Seconds

If using these identifiers in the file name, they must be included within curly braces.

For example, a value of "MyCustomReport{yyyyMMdd}" generates an output file name similar to "MyCustomReport20160101".

The default value for this field is <ReportType>-yyyyMMddhhmmss.

4.2.8   IncludeHeaders

Setting IncludeHeaders to TRUE includes the column headers as the first row of the returned report while setting the field to FALSE does not include a header row. This setting applies to the NBA and NCA files as well if generated.

The default value for this field is 0.

4.2.9   Precision

The Precision field defines the number of places a decimal shall return.

The default value for this field is 0.

4.2.10   ReportType

The ReportType field specifies whether to use a pre-defined Avalara report template or if the report shall be a custom report. For further information, see 4.3 – Report Types.

This field is required.

4.2.11 StartDate, EndDate, and TimeSpan

The data to be included in the report is mainly specified by the StartDate and EndDate fields of the ReportOptions parameter that is passed in to the CreateReport API. The dates are compared against the server date at the time when the transaction was submitted and processed by the web service.

The StartDate may not be more than 90 days in the past. The EndDate or TimeSpan may not be more than 31 days apart from the StartDate. The EndDate is non-inclusive, so only records with a timestamp smaller than the EndDate will be included in the report. For example, to include all the data for January 2016 in the report, set the StartDate to "01/01/2016" and the EndDate to "02/01/2016".

StartDate must be set but either EndDate or TimeSpan may be used. If both EndDate and TimeSpan are set, CreateReport uses the TimeSpan value.

StartDate is a required field. Either EndDate or TimeSpan is required.

4.3   Report Types

The following report types are available. Note that each of the reports specified below will be generated as CSV files.

Report Type	Description
srtcomma20l	Generates report output similar to the srtcomma20l utility in AFC License. The columns included are: CountryIso, State, County, Locality, TaxType, TaxLevel, Rate, TaxAmount, SaleAmount, ExemptSaleAmount, RefundUncollect, NetTaxableMeasure, Minutes, and Lines. Please refer to **TM_00523_srtcomma20l.pdf** for additional information.
srtcomma20l-p	Generates report output similar to the srtcomma20l utility in AFC License using the –p option to output PCodes instead of the location names. The columns included are: PCode, TaxType, TaxLevel, Rate, TaxAmount, SaleAmount, ExemptSaleAmount, RefundUncollect, NetSaleAmount, Minutes, and Lines. Please refer to **TM_00523_srtcomma20l.pdf** for additional information.
srtcomma20ld	Generates report output similar to the srtcomma20ld utility in AFC License. The columns included are: CountryIso, State, County, Locality, TaxType, TaxDescription, TaxLevel, TaxLevelDesc, Rate, TaxAmount, GrossSales, ExemptSaleAmount, RefundUncollect, NetTaxableMeasure, Minutes, and Lines.

Report Type	Description
	Please refer to **TM_00524_srtcomma20ld.pdf** for additional information.
srtcommadetail	Generates report output similar to the srtcommadetail utility in AFC License. The columns included are: CountryIso, State, County, Locality, TaxType, TaxLevel, DiscountType, CalcType, Rate, TaxAmount, GrossSales, ExemptSaleAmount, RefundUncollect, NetSaleAmount, Minutes. Please refer to **TM_00525_ srtcommadetail.pdf** for additional information.
customlog	A custom report containing the predefined columns of the report type specified in the BaseReport field of the ReportOptions and/or the fields specified in the CustomLogField array.

4.4   Custom Log Report Columns

The following columns are available when generating a report using the customlog ReportType. For each column, the Include and Sort proprieties must also be specified.

A Calculated column refers to data calculated within and returned from the AFC Engine.

Column Name	Description	Calculated
CountryISO	Three-character country code for taxing jurisdiction.	No
State	State abbreviation for taxing jurisdiction.	No
County	County name for taxing jurisdiction.	No
Locality	Locality name for taxing jurisdiction.	No
CompanyIdentifier	Input entered in the CompanyIdentifier field of the TelecomTransaction or SalesUseTansaction.	No
CustomerNumber	Input entered in the CustomerNumber field of the TelecomTransaction or SalesUseTansaction.	No
Date	Input entered in the Date field of the TelecomTransaction or SalesUseTansaction.	No
DiscountType	Input entered in the DiscountType field of the TelecomTransaction or SalesUseTansaction.	No
DiscountTypeDesc	Description for DiscountType field.	No
ExemptionType	Input entered in the ExemptionType field of the TelecomTransaction or SalesUseTansaction.	No
ExemptionTypeDesc	Description for ExemptionType field.	No
InvoiceNumber	Input entered in the InvoiceNumber field of the TelecomTransaction or SalesUseTansaction.	No
Optional	Input entered in the Optional field of the TelecomTransaction or SalesUseTansaction.	No
Optional4	Input entered in the Optional4 field of the TelecomTransaction or SalesUseTansaction.	No

Column Name	Description	Calculated
Optional5	Input entered in the Optional5field of the TelecomTransaction or SalesUseTansaction.	No
Optional6	Input entered in the Optional6 field of the TelecomTransaction or SalesUseTansaction.	No
Optional7	Input entered in the Optional7 field of the TelecomTransaction or SalesUseTansaction.	No
Optional8	Input entered in the Optional8 field of the TelecomTransaction or SalesUseTansaction.	No
Optional9	Input entered in the Optional9 field of the TelecomTransaction or SalesUseTansaction.	No
Optional10	Input entered in the Optional10 field of the TelecomTransaction or SalesUseTansaction.	No
OptionalAlpha1	Input entered in the OptionalAlpha1 field of the TelecomTransaction or SalesUseTansaction.	No
ServiceLevelNumber	Input entered in the ServiceLevelNumber field of the TelecomTransaction or SalesUseTansaction.	No
Billable	Billable flag for tax.	No
CalcType	Calculation type for tax.	No
CalcTypeDesc	Description for CalcType field.	No
CategoryID	Category ID for tax.	No
CategoryDesc	Category description for tax.	No
ExemptSaleAmount	Exempt sale amount from tax record.	No
Compliance	Compliance flag from tax record.	No
PCode	PCode for taxing jurisdiction	No
Rate	Rate for tax.	No
RefundUncollect	For adjustments, this is the refunded taxable measure from tax record.	Yes
ServiceType	Service Type used in the input transaction or derived out of the bundle if a bundle transaction was used.	No
Surcharge	Surcharge flag from tax record.	No
TaxableMeasure	Taxable measure used for calculating the tax.	Yes
SaleAmount	Taxable measure plus exempt sale amount (typically the input sale amount entered in the transaction. This field is used for srtcomma20l and srtcomma20l-p reports.	Yes
GrossSales	Gross sale amount. This field is used for srtcomma20ld and srtcommadetail reports.	Yes
NetSaleAmount	Net sale amount. This field is used for srtcomma20ld and srtcommadetail reports.	Yes
NetTaxableMeasure	Net taxable measure. This field is used for srtcomma20l and srtcomma20l-p reports.	Yes
TaxAmount	Tax amount.	Yes
TaxLevel	Tax level identifier.	No
TaxLevelDesc	Description for tax level identifier.	No
TaxType	Tax type identifier.	No
TaxDescription	Description for tax type.	No
TransactionID	Transaction ID used in the input transaction or derived out of the bundle if a bundle transaction was used.	No

Column Name	Description	Calculated
ServiceID	Service ID used in the input transaction or derived out of the bundle if a bundle transaction was used.	No
Lines	Number of lines used for calculating the tax in per-line taxes.	Yes
Locations	Number of locations	Yes
Minutes	Number of minutes used for calculating the tax in per-minute taxes.	Yes
TaxLogID	Primary key identifier for transaction.	No
ServerDate	Server date when the transaction was processed.	No
OptionalField1 OptionalField2 OptionalField3 OptionalField4 OptionalField5 OptionalField6 OptionalField7 OptionalField8 OptionalField9 OptionalField10	Additional Extended Optional fields in which column headers are defined by the user upon creation of each one.	No

4.5 Aggregation

Any records included in the report will be grouped together using any non-calculated fields that are included in the output. The non-calculated field values are aggregated. To avoid any grouping, include the TaxLogID column in the report output. For a list of calculated and non-calculated fields, see 4.3 - Custom Log Report Columns.

4.6 Output Files

The output files are compressed into a zip file with the name as specified in the FileName field. This zip file is then placed in the specific client FTP folder. The zip file may contain multiple files if the CreateNbaFile or CreateNcaFile options are set to true in the ReportOptions.

4.6.1 Accessing the FTP Site

AFC SaaS Pro provides two ways to connect to the AFC FTP server - by using a secure file-transfer protocol (SFTP) or file-transfer protocol (FTP) connection through an existing account and typing 'ftp.billsoft.com' at the prompt within the FTP session or by connecting to the web interface at https://ftp.billsoft.com.

All client FTP sites must be set up properly and clients must have their username and password. Please contact communicationsupport@avalara.com to obtain this information or for assistance with accessing output files.

Note Concerning FTP Client Software

Please be aware that web browsers (Microsoft Internet Explorer, Firefox, etc.) cannot be used for uploading and downloading files via FTP to the AFC FTP site. A FTP client application must be used to transfer files to and from the AFC FTP site when using our service as FTP.

Windows users:

Microsoft does not currently include convenient FTP client software in its operating systems. It is assumed you either own a third-party FTP client application such as WS_FTP (a shareware version of which can be downloaded from http://www.ipswitch.com), or that you are comfortable accessing FTP sites using command-line syntax.

Linux users:

If using a Linux system to transfer your call-data files, you can use any number of free FTP clients to contact the AFC FTP site, such as WXFTP, or you can use command-line syntax.

4.6.1.1 Logging on to the FTP Site

To log on to the AFC FTP site, you will need the following information:

HOSTNAME/ADDRESS: 'ftp.billsoft.com' within a FTP session or via the web at https://ftp.billsoft.com

USER ID: Your User ID is created and provided to you. Please contact communicationsupport@avalara.com for your organization's assigned User ID.

PASSWORD: Your organization's login password is also provided by your Avalara technical support contact. For future reference, you may make note of your login details in spaced provided below.

User ID: _____

Password: _____

Company Code: _____
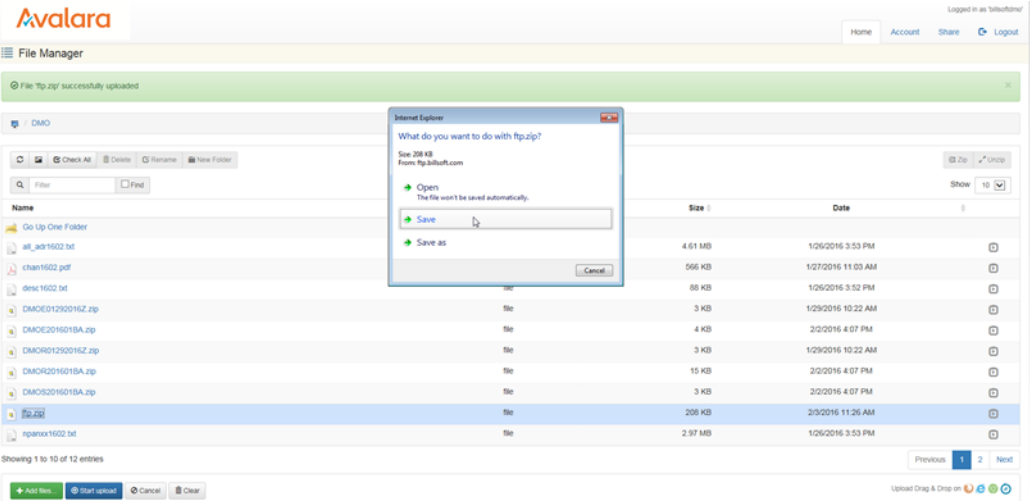
4.6.1.2 Downloading Files from the FTP Site

Once you are connected to the AFC FTP site, you should change to the directory that has the same name as your company's three-character code.

When the report is available, you can either download the output from the AFC FTP or get it from the https://ftp.billsoft.com site.

To download from the AMC FTP site, connect within a FTP session, open the folder with you company's three-character code, and select the file you wish to download. Transfer the selected file to your destination using the tool's transfer method.

To download from the https://ftp.billsoft.com web interface, follow the steps below:

Step	Action/Result
1	View the completed files processed and locate the file with appropriate date and timestamp.
2	Highlight or double-click the file you wish to open.
3	Select the **Save** or **Save as** option in the window. **Note:** The file must be saved. If the file is opened and closed without being saved, it will no longer be available on the FTP site to download and save for your records.

4.7   Sample Code

```
// Create instance of web service client
        var client = new EZTaxWebServiceClient("BasicHttpBinding_IEZTaxWebService");
        client.ClientCredentials.UserName.UserName = "(MyUserName)";
        client.ClientCredentials.UserName.Password = "(MyPassword)";

        // Create a ReportOptions object to specify the options for the report
        var reportOptions = new ReportOptions
        {
            BaseReport = "srtcomma20l",
            CreateNbaFile = true,
            CreateNcaFile = false,
            EmailAddress = "myemail@address.com",
            EndDate = new DateTime(2016, 5, 1),
            FileExtension = ".csv",
            FileName = "test-{yyyyMMdd}",
            IncludeHeaders = true,
            Precision = 6,
            ReportType = "customlog",
            StartDate = new DateTime(2016, 4, 1)
        };


// For customlog reports, you may specify individual fields to be included
        reportOptions.CustomLogFields = new[]
        {
            new CustomLogField
            {
                Column = "PCode",
                Include = true,
                Sort = false
            },
            new CustomLogField
            {
                Column = "CategoryID",
                Include = true,
                Sort = true
            },
            new CustomLogField
            {
                Column = "CategoryDesc",
                Include = true,
                Sort = false
            }
        };

        try
        {

// Invoke the CreateReport API to submit the report for processing
            client.CreateReport(reportOptions);
            Console.WriteLine("Report submitted for processing.");
        }
        catch (FaultException ex)
        {
            Console.WriteLine("An error occurred submitting the report: " + ex.ToString());
        }
```

5. Bridge Conferencing

The Bridge Conferencing feature is currently in development at this time. Please refrain from using this feature as well as the proposed solutions and APIs until further notice.

6. Optional Fields

6.1 Overview

AFC SaaS Pro provides a number of optional fields for use in reports.

There are two types of Optional Fields available in AFC SaaS Pro- 9 default Optional Fields and up to 10 Extended Optional Fields. All Optional Fields are part of the transaction.

Note: The Optional Fields do not impact taxation.

6.2 Default Optional Fields

There are 9 default Optional Fields.

Reporting Fields	
Column Name	**Description**
Optional	Unsigned int. Should be >= 0. The max value is different for different platforms, but in general supports values at least up to 4294967295 (2^{32} -1).
Optional4	
Optional5	
Optional6	
Optional7	
Optional8	
Optional9	
Optional10	
OptionalAlpha1	Alpha-numeric field. Supports up to 20 bytes.

Note: Please refer to Sections **1.9 Compliance Files** and **1.11 Compliance Month** for additional information regarding the Optional10 field.

6.3 Extended Optional Fields

AFC SaaS Pro has the ability to pass up to 10 additional Extended Optional Fields. The general process is as follows:

1. Add the Extended Optional Fields using the OptionalField property of the Transaction.
2. Update the description of a specified Extended Optional Field by invoking the UpdateOptionalFieldKeyDesc API.
3. Display a list of all updated descriptions by invoking the GetOptionalFieldKeyDesc API.

The Extended Optional Fields are not currently supported in Single-Transaction Invoice Mode or Batch-Transaction Invoice Mode.

Note: The Extended Optional Fields passed as part of the Transaction are persisted until otherwise changed. For example, if the first OptionalField is used in a transaction and updated with a description of "Purchase Order Number", this field should continue to be utilized as a Purchase Order Number until at least the end of the reporting cycle so that mixed data does not appear on the resulting Detail Log report or Custom Report.

6.3.1   Setting Up an Extended Optional Field in a Transaction

A Transaction containing one or more Extended Optional Fields is set up as usual with the addition of the OptionalField property. Each OptionalField specifies an OptionalKeyNo and an OptionalValue.

Field	Description
OptionalKeyNo	Optional field key number (integer values from 1 to 10). The OptionalKeyNo field is the key associated with the corresponding value.
OptionalValue	Value for optional field (up to 150 characters). The OptionalValue field is the value that will display in the body of the reports.

Note: To avoid possible issues with CSV reports, it is advised that the use of special characters, especially the comma (,) should be avoided.

6.3.2   UpdateOptionalFieldKeyDesc

The UpdateOptionalFieldKeyDesc API is used to update the description for each optional field key thereby indicating what each field in the OptionalField array of the Transaction is used for (e.g. invoice number, line item number, etc.). The values specified will be used when generating reports that contain these optional fields.

To update a description, pass the **OptionalKeyNo** that should be updated as well as the desired description in the **OptionalValue** field.

Field	Description
OptionalKeyNo	Optional field key number (integer values from 1 to 10). The OptionalKeyNo field is the key originally passed as part of the transaction.
OptionalValue	Value for the optional field description (up to 150 characters). The OptionalValue field is the value of the description of the specified Optional Field that will display on the reports.

Note: To avoid possible issues with CSV reports, it is advised that the use of special characters, especially the comma (,) should be avoided.

6.3.3　GetOptionalFieldKeyDesc

The GetOptionalFieldKeyDesc API returns the description for each optional field key in the OptionalField array of the Transaction that has been updated using the UpdateOptionalFieldKeyDesc API. The description values specified will be used when generating reports that contain these optional fields.

7. Zip Lookup Requests

7.1　Overview

Zip Lookup functionality returns all or multiple PCodes and jurisdiction details associated with the address input provided.

To use this functionality, invoke the ZipLookup API.

7.2　Setting up a Request

In order to call the API, the ZipLookup data structure needs to be populated. At least one location field is required. If Country is not specified, the default value is set to "USA".

7.2.1　Best Match

By default the ZipLookup algorithm requires an exact match of all fields provided with the following considerations:

1. Punctuation in city names will be ignored
2. Whitespace in city names will be ignored. For example, the city name LAND O' LAKES will match LAND O LAKES or LANDOLAKES as both the punctuation and spacing will be ignored for purposes of matching.

If Best Match is set to true, some limited variable search algorithms will be used to find the best match for the data provided.

Setting the **BestMatchFlag** to TRUE

- Best Match first attempts an exact match

- If and only if the exact match search fails, the best match algorithm will be applied to find matches based on variable search algorithms.

Setting the **BestMatchFlag** to FALSE produces exact matches only.

- Produces exact matches only

The results field **MatchTypeApplied** indicates whether Exact match or Best match was used.

7.2.2   Location Data

At least one location field is required. If Country is not specified, the default value is set to "USA".

Specify a location by providing one or more of the following fields: **City**, **Country**, **County**, **State**, **ZipCode**.

7.2.3   Limit Results

The **LimitResults** field is used to limit the number of matches returned.

The default value is 100 while the maximum limit is 1,000. If 0 is entered as a value, the default limit is used. If the value entered is greater than the maximum limit, the maximum limit is used.

7.3   Results

The LocationData structure is returned for each location found based on the provided input. All results are contained within the ZipLookupResult structure.

7.4   Examples

Examples using Manhattan in New York are as follows:

1. Sample Request with one exact match

    Request 1 - Match exact on Country, State, County, City and Zip Code

Country	State	County	Locality	ZipCode	Best Match
USA	NY	New York	Manhattan	10001	FALSE

Result

PCode	Country	State	County	City
2604100	USA	NY	NEW YORK	MANHATTAN

2. Sample Request with multiple exact matches

Request 1 – Match exact on Country, State, County and Zip Code

Country	State	County	Locality	ZipCode	Best Match
USA	NY	New York		10001	FALSE

Request 2 – Match exact on Country and Zip Code (Blank Country defaults to USA)

Country	State	County	Locality	ZipCode	Best Match
				10001	FALSE

Result - Both match 5 addresses (All have same Country, State, County and Zip)

PCode	Country	State	County	City
2604100	USA	NY	NEW YORK	EMPIRE STATE
2604100	USA	NY	NEW YORK	GREELEY SQUARE
2604100	USA	NY	NEW YORK	ONE HUNDRED THIRTY EIGHTH
2604100	USA	NY	NEW YORK	MANHATTAN
2604100	USA	NY	NEW YORK	NEW YORK

3. Sample Request – Match exact on invalid address

Request 1 – Match exact on Country, State, County, City and Zip Code

Country	State	County	Locality	ZipCode	Best Match
USA	NY	Westchester	Manhattan	10001	FALSE

Result – No matches, not a valid address

PCode	Country	State	County	City

4. Sample Request – Best match on invalid address

Request 1 – Best match on Country, State, County, City and Zip Code

Country	State	County	Locality	ZipCode	Best Match
USA	NY	Westchester	Manhattan	10001	FALSE

Result – County is disregarded

PCode	Country	State	County	City
2604100	USA	NY	NEW YORK	MANHATTAN

8. Safe Harbor Overrides for Traffic Studies

The results of percentages from a traffic study can be applied by administering traffic study TAM overrides. The TAM values for Cellular, VoIP and Paging are adjusted to reflect the percentages in the traffic study.

Please contact communicationsupport@avalara.com in order to set the override at the account level versus transaction level.

1. Cellular Examples:

Original Cellular Safe Harbor: 37.1% Federal / 62.9% State											
State	City	Zip	Charge	Products	Tax Type	Level	Rate	TM	Tax	Charge	Exempt
NY	Manhattan	10001	100	Cellular/Access Charge	Fed USF Cellular	Federal	0.174	37.1	6.4554	100	62.9

Cellular Traffic Study Override: 15.0% Federal / 85.0% State											
State	City	Zip	Charge	Products	Tax Type	Level	Rate	TM	Tax	Charge	Exempt
NY	Manhattan	10001	100	Cellular/Access Charge	Fed USF Cellular	Federal	0.174	15	2.61	100	85

2. VoIP Examples:

Original VoIP Safe Harbor: 64.9% Federal / 35.1% State										
State	Zip	Charge	Products	Tax Type	Level	Rate	TM	Tax	Charge	Exempt
CA	90001	100	VoIP/Access Charge	Universal Lifeline Telephone Service Charge (VoIP)	State	0.055	35.1	1.9305	100	64.9
CA	90001	100	VoIP/Access Charge	CASF (VoIP)	State	0.00464	35.1	0.16286	100	64.9
CA	90001	100	VoIP/Access Charge	CA Teleconnect Fund (VoIP)	State	0.0108	35.1	0.37908	100	64.9
CA	90001	100	VoIP/Access Charge	CA High Cost Fund A (VoIP)	State	0.0035	35.1	0.12285	100	64.9
CA	90001	100	VoIP/Access Charge	TRS (VoIP)	State	0.005	35.1	0.1755	100	64.9
CA	90001	100	VoIP/Access Charge	E911 (VoIP)	State	0.0075	35.1	0.26325	100	64.9

Original VoIP Safe Harbor: 64.9% Federal / 35.1% State										
CA	90001	100	VoIP/Access Charge	FUSF (VoIP)	Federal	0.174	64.9	11.2926	100	35.1
CA	90001	100	VoIP/Access Charge	FCC Regulatory Fee (VoIP)	Federal	0.00371	64.9	0.24078	100	35.1

VoIP Traffic Study Override: 25.0% Federal / 75.0% State										
State	Zip	Charge	Products	Tax Type	Level	Rate	TM	Tax	Charge	Exempt
CA	90001	100	VoIP/Access Charge	Universal Lifeline Telephone Service Charge (VoIP)	State	0.055	75	4.125	100	25
CA	90001	100	VoIP/Access Charge	CASF (VoIP)	State	0.00464	75	0.348	100	25
CA	90001	100	VoIP/Access Charge	CA Teleconnect Fund (VoIP)	State	0.0108	75	0.81	100	25
CA	90001	100	VoIP/Access Charge	CA High Cost Fund A (VoIP)	State	0.0035	75	0.2625	100	25
CA	90001	100	VoIP/Access Charge	TRS (VoIP)	State	0.005	75	0.375	100	25
CA	90001	100	VoIP/Access Charge	E911 (VoIP)	State	0.0075	75	0.5625	100	25
CA	90001	100	VoIP/Access Charge	FUSF (VoIP)	Federal	0.174	25	4.35	100	75
CA	90001	100	VoIP/Access Charge	FCC Regulatory Fee (VoIP)	Federal	0.00371	25	0.09275	100	75

9. AFC SaaS Pro Telecom Web Service Programmer Reference

The AFC SaaS Pro Telecom Web Service was developed using XML, SOAP 1.1, and WSDL so it can be integrated into virtually any application. To use the web service from your application, you will need to create a proxy stub for your programming language and platform. The proxy stub will encapsulate many of the details of communicating over the Internet between your application and the web service. The proxy stub will contain data types, classes and functions that you will use in your source code to invoke the methods on the web service.

Most programming languages have a toolkit or SDK for generating some or all of the proxy stub. The following is a list of some of the products that may be used to create the proxy stub.

Toolkit or Product Name	Programming Language	Platform
Visual Studio.NET	C# or Visual Basic.NET	Microsoft Windows
.NET Framework SDK	C# or Visual Basic.NET	Microsoft Windows
Systinet Server	Java or C++	See http://www.systinet.com
Apache Axis	Java or C++	See http://ws.apache.org/axis/
GSOAP	C/C++	See http://sourceforge.net/projects/gsoap2

9.1   TaxService Endpoint

TaxService WSDL (secured with user id and password in SOAP header) Avalara recommends using HTTPS when connecting to this endpoint. The URL for this endpoint is http://EZtaxasp.billsoft.com/EZtaxWebService/EZtaxWebService.svc?wsdl.

There will be two ways to access the service, either via https or http. The TaxService endpoint requires the user id and password to be passed in a custom SOAP header element. These are sent in clear text, so Avalara requires the client to use one of the security models to secure the password.

Common Properties

WSDL URL   https://eztaxasp.billsoft.com/EZTaxWebService/EZTaxWebService.svc?wsdl

Namespace   http://tempuri.org/

Binding Specific Properties

Binding	Endpoint	Soap Version
BasicHttpBinding_IEZTax WebService	https://eztaxasp.billsoft.com/EZTaxWebService/EZTaxWebService.svc	SOAP 1.1
BasicHttpBinding_IEZTax WebService1	https://eztaxasp.billsoft.com/EZTaxWebService/EZTaxWebService.svc/Soap11	SOAP 1.1
CustomBinding_IEZTaxW ebService	https://eztaxasp.billsoft.com/EZTaxWebService/EZTaxWebService.svc/SSL	SOAP 1.2

Deprecated Binding

Binding	Endpoint	Soap Version
WSHttpBinding_IEZTaxW ebService	http://eztaxasp.billsoft.com/EZTaxWebService/EZTaxWebService.svc	SOAP 1.2

The remainder of this document describes the web service methods and customer data types required to call AFC SaaS Pro from your application. With this interface, the applications pass the transaction data to AFC SaaS Pro as they are being sold or billed. AFC SaaS Pro calculates all required taxes and returns the tax information to the billing system per transaction.   In addition, AFC SaaS Pro stores all tax data generated in a database. This data can then be used to generate reports used for tax compliance. AFC SaaS Pro is capable of generating files that can be electronically processed by Atlantax, Ernst & Young, and Tax Partners, Inc., for tax compliance filing. AFC SaaS Pro provides facilities to generate tax adjustments or refunds based upon un-collectable accounts or customer refunds. Adjustment

information is returned to the application and is utilized to update tax data for report generation and compliance filing.

10.SOAP Telecom Web Methods

These methods process each transaction independently.

Any of these messages can produce an exception.   Errors can occur in any of the calculation methods are listed in Error Messages Common to all Calculation Methods. Messages that are specific to a particular method are shown with each method description.

Method Name	Summary
CalcTaxesWithPCode	Calculate taxes on supplied telecom transaction using PCodes
CalcTaxesWithNpaNxx	Calculate taxes on supplied telecom transaction using NpaNxx values
CalcTaxesWithZipAddress	Calculate taxes on supplied telecom transaction using Zip addresses
CalcTaxesWithFipsCode	Calculate taxes on supplied telecom transaction using FIPS Codes
CalcAdjWithPCode	Calculated adjustment on supplied transaction using PCodes
CalcAdjWithNpaNxx	Calculated adjustment on supplied transaction using NpaNxx values
CalcAdjWithZipAddress	Calculated adjustment on supplied transaction using Zip addresses
CalcAdjWithFipsCode	Calculated adjustment on supplied transaction using FIPS Codes
CalcReverseTaxesWithPCode*	Performs a tax inclusive calculation to arrive at the base sale amount and taxes for the desired total charge using PCodes.
CalcReverseTaxesWithFipsCode*	Performs a tax inclusive calculation to arrive at the base sale amount and taxes for the desired total charge using Fips Codes.
CalcReverseTaxesWithZipAddress*	Performs a tax inclusive calculation to arrive at the base sale amount and taxes for the desired total charge using Zip Address.
CalcReverseTaxesWithNpaNxx*	Performs a tax inclusive calculation to arrive at the base sale amount and taxes for the desired total charge using NPANXX.
CalcReverseAdjWithPCode*	Performs a tax inclusive adjustment to arrive at the base sale amount and taxes for the desired total charge using PCodes.
CalcReverseAdjWithFipsCode*	Performs a tax inclusive adjustment to arrive at the base sale amount and taxes for the desired total charge using Fips Codes.
CalcReverseAdjWithZipAddress*	Performs a tax inclusive adjustment to arrive at the base sale amount and taxes for the desired total charge using Zip Address.
CalcReverseAdjWithNpaNxx*	Performs a tax inclusive adjustment to arrive at the base sale amount and taxes for the desired total charge using NPANXX.
CalcTaxInclusiveTaxesWithPCode	Performs a tax inclusive calculation to arrive at the base sale amount and taxes for the desired total charge using PCodes.
CalcTaxInclusiveTaxesWithFipsCode	Performs a tax inclusive calculation to arrive at the base sale amount and taxes for the desired total charge using Fips Codes.

Method Name	Summary
CalcTaxInclusiveTaxesWithZipAddress	Performs a tax inclusive calculation to arrive at the base sale amount and taxes for the desired total charge using Zip Address.
CalcTaxInclusiveTaxesWithNpaNxx	Performs a tax inclusive calculation to arrive at the base sale amount and taxes for the desired total charge using NPANXX.
CalcTaxInclusiveAdjWithPCode	Performs a tax inclusive adjustment to arrive at the base sale amount and taxes for the desired total charge using PCodes.
CalcTaxInclusiveAdjWithFipsCode	Performs a tax inclusive adjustment to arrive at the base sale amount and taxes for the desired total charge using Fips Codes.
CalcTaxInclusiveAdjWithZipAddress	Performs a tax inclusive adjustment to arrive at the base sale amount and taxes for the desired total charge using Zip Address.
CalcTaxInclusiveAdjWithNpaNxx	Performs a tax inclusive adjustment to arrive at the base sale amount and taxes for the desired total charge using NPANXX.
BeginCustomerBatch	Flags the beginning of a customer transaction batch and returns a customer batch ID that can be used to submit each transaction and process the entire batch.
CalcCustTaxes	Submits a transaction for the specified customer batch.
CalcCustAdj	Submits an adjustment for the specified customer batch.
ProcessCustomerBatch	Processes a customer batch and returns the tax calculation results.
CalcTaxesInCustMode	Processes a batch of telecom and/or sales and use transactions and adjustments in Invoice Mode.
CalcProRatedTaxes	Accepts a pro-rated percentage that is used to calculate the taxable amount on the transaction and perform tax calculations.
CalcProRatedAdj	Accepts a pro-rated percentage that is used to calculate the taxable amount on the transaction and perform tax adjustments.
CalcJurisdiction	Determines the taxing jurisdiction for a transaction and returns the PCode at the lowest jurisdiction level.
CalcTaxesWithOverrides	Accepts transaction data and tax rate override data to perform tax calculations with provided override information.
CalcAdjWithOverrides	Accepts transaction data and tax rate override data to perform tax adjustment calculations with provided override information.
CalcBridgeConferenceTaxes	Calculate taxes on supplied telecom transaction using sourcing rules for conferencing. **Note:** *The Bridge Conferencing feature is currently in development at this time. Please refrain from using this feature as well as the proposed solutions and APIs until further notice.*
CalcAdjBridgeConferenceTaxes	Calculated adjustment on supplied transaction using sourcing rules for conferencing. **Note:** *The Bridge Conferencing feature is currently in development at this time. Please refrain from using this feature as well as the proposed solutions and APIs until further notice.*

Method Name	Summary
CommitTransactions	Used to commit or uncommit a DocumentCode.

It is recommended to begin using the newly renamed 'tax inclusive' APIs as the 'reverse' APIs will be deprecated in the future.

10.1 CalcTaxesWithPCode

This method accepts transaction data and performs appropriate tax calculations. Origination, Termination, and Bill To information is passed using PCodes.

Return Type:

TaxData[]   An array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:
- OriginationPCode must be set! – The OriginationPCode property of the Transaction was not set
- TerminationPCode must be set! – The TerminationPCode property of the Transaction was not set
- BillToPCode must be set! – The BillToPCode property of the Transaction was not set
- Transaction is null! – No Transaction passed in
- PCode not found. – One of the three PCode values passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
None.

10.2 CalcTaxesWithNpaNxx

This method accepts transaction data and performs appropriate tax calculations. Origination, Termination, and Bill To information is passed using NpaNxx values.

Return Type:

TaxData[]   An array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:

- OriginationNpaNxx must be set! – The OriginationNpaNxx property of the Transaction was not set
- TerminationNpaNxx must be set! – The TerminationNpaNxx property of the Transaction was not set
- BillToNpaNxx must be set! – The BillToNpaNxx property of the Transaction was not set
- Transaction is null! – No Transaction passed in
- NPANXX not found. – One of the three NpaNxx values passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
None.

10.3 CalcTaxesWithZipAddress

This method accepts transaction data and performs appropriate tax calculations. Origination, Termination, and Bill To information is passed using Zip Addresses.

Return Type:

TaxData[]   An array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:

- OriginationZipAddress must be set! – The OriginationZipAddress property of the Transaction was not set
- TerminationZipAddress must be set! – The TerminationZipAddress property of the Transaction was not set
- BillToZipAddress must be set! – The BillToZipAddress property of the Transaction was not set
- Transaction is null! – No Transaction passed in
- County/State/Zip not found. – One of the three ZipAddress values passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
None.

10.4 CalcTaxesWithFipsCode

This method accepts transaction data and performs appropriate tax calculations. Origination, Termination, and Bill To information is passed using FIPS Code values.

Return Type:

TaxData[]	An array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction	Telecom transaction data

Errors:
- OriginationFipsCode must be set! – The OriginationFipsCode property of the Transaction was not set.
- TerminationFipsCode must be set! – The TerminationFipsCode property of the Transaction was not set.
- BillToFipsCode must be set! – The BillToFipsCode property of the Transaction was not set.
- Transaction is null! – No Transaction passed in
- FIPS not found. – One of the three FipsCode values passed in is not in AFC SaaS Pro
- FIPS (to PCode) cross-reference database not open. - – Indicates an error occurred on the server

See also Error Messages Common to all Calculation Methods
Remarks:
None.

10.5 CalcAdjWithPCode

This method accepts transaction data and performs appropriate tax adjustment calculations. Origination, Termination, and Bill To information is passed using PCodes.

Return Type:

TaxData[]	An array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction	Telecom transaction data

Errors:
- OriginationPCode must be set! – The OriginationPCode property of the Transaction was not set
- TerminationPCode must be set! – The TerminationPCode property of the Transaction was not set
- BillToPCode must be set! – The BillToPCode property of the Transaction was not set
- Adjustment is null! – No Adjustment passed in
- PCode not found. – One of the three PCode properties passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

Discounts can also be processed by setting the DiscountType field in the adjustment transaction.

10.6 CalcAdjWithNpaNxx

This method accepts transaction data and performs appropriate tax adjustment calculations. Origination, Termination, and Bill To information is passed using NpaNxx values.

Return Type:

TaxData[]   An array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:

- OriginationNpaNxx must be set! – The OriginationNpaNxx property of the Transaction was not set.
- TerminationNpaNxx must be set! – The TerminationNpaNxx property of the Transaction was not set.
- BillToNpaNxx must be set! – The BillToNpaNxx property of the Transaction was not set.
- Adjustment is null! – No Adjustment passed in
- NPANXX not found. – One of the three NpaNxx properties passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:

Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

Discounts can also be processed by setting the DiscountType field in the adjustment transaction.

10.7 CalcAdjWithZipAddress

This method accepts transaction data and performs appropriate tax adjustment calculations. Origination, Termination, and Bill To information is passed using Zip addresses.

Return Type:

TaxData[]   An array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:

- OriginationZipAddress must be set! – The OriginationZipAddress property of the Transaction was not set.

- TerminationZipAddress must be set! – The TerminationZipAddress property of the Transaction was not set.
- BillToZipAddress must be set! – The BillToZipAddress property of the Transaction was not set.
- Adjustment is null! – No Adjustment passed in
- County/State/Zip not found. – One of the three ZipAddress values passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

Discounts can also be processed by setting the DiscountType field in the adjustment transaction.

10.8 CalcAdjWithFipsCode

This method accepts transaction data and performs appropriate tax adjustment calculations. Origination, Termination, and Bill To information is passed using FIPS Codes.

Return Type:

TaxData[]   An array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:
- OriginationFipsCode must be set! – The OriginationFipsCode property of the Transaction was not set.
- TerminationFipsCode must be set! – The TerminationFipsCode property of the Transaction was not set.
- BillToFipsCode must be set! – The BillToFipsCode property of the Transaction was not set.
- Adjustment is null! – No Adjustment passed in
- FIPS no found. – One of the three FIPS codes passed in is not in AFC SaaS Pro
- FIPS (to PCode) cross-reference database not open. – Indicates an error occurred on the server

See also Error Messages Common to all Calculation Methods

Remarks:
Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

Discounts can also be processed by setting the DiscountType field in the adjustment transaction.

10.9 CalcReverseTaxesWithPCode

It is recommended to begin using the newly renamed 'tax inclusive' API, CalcTaxInclusiveTaxesWithPCode, as the 'reverse' API will be deprecated in the future.

This method accepts transaction data and performs a tax inclusive calculation in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using PCodes.

Return Type:

ReverseTaxResults   Contains the calculated base sale amount for the transaction and an array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:

- OriginationPCode must be set! – The OriginationPCode property of the Transaction was not set
- TerminationPCode must be set! – The TerminationPCode property of the Transaction was not set
- BillToPCode must be set! – The BillToPCode property of the Transaction was not set
- Transaction is null! – No Transaction passed in
- PCode not found. – One of the three PCode values passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:

The desired total charge must be placed in the Charge field of the Transaction.

10.10 CalcReverseTaxesWithFipsCode

It is recommended to begin using the newly renamed 'tax inclusive' API, CalcTaxInclusiveTaxesWithFipsCode, as the 'reverse' API will be deprecated in the future.

This method accepts transaction data and performs a tax inclusive calculation in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using Fips Codes.

Return Type:

ReverseTaxResults   Contains the calculated base sale amount for the transaction and an array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:

- OriginationFipsCode must be set! – The OriginationFipsCode property of the Transaction was not set.
- TerminationFipsCode must be set! – The TerminationFipsCode property of the Transaction was not set.
- BillToFipsCode must be set! – The BillToFipsCode property of the Transaction was not set.
- Transaction is null! – No Transaction passed in
- FIPS no found. – One of the three FIPS codes passed in is not in AFC SaaS Pro
- FIPS (to PCode) cross-reference database not open. – Indicates an error occurred on the server

See also Error Messages Common to all Calculation Methods

Remarks:
The desired total charge must be placed in the Charge field of the Transaction.

10.11 CalcReverseTaxesWithZipAddress

*It is recommended to begin using the newly renamed 'tax inclusive' API, CalcTaxInclusiveTaxesWithZipAddress, as the 'reverse' API will be deprecated in the future.

This method accepts transaction data and performs a tax inclusive calculation in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using ZipAddress objects.

Return Type:

ReverseTaxResults	Contains the calculated base sale amount for the transaction and an array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction	Telecom transaction data

Errors:
- OriginationZipAddress must be set! – The OriginationZipAddress property of the Transaction was not set.
- TerminationZipAddress must be set! – The TerminationZipAddress property of the Transaction was not set.
- BillToZipAddress must be set! – The BillToZipAddress property of the Transaction was not set.
- Transaction is null! – No Transaction passed in
- County/State/Zip not found. – One of the three ZipAddress values passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
The desired total charge must be placed in the Charge field of the Transaction.

10.12 CalcReverseTaxesWithNpaNxx

It is recommended to begin using the newly renamed 'tax inclusive' API, CalcTaxInclusiveTaxesWithNpaNxx, as the 'reverse' API will be deprecated in the future.

This method accepts transaction data and performs a tax inclusive calculation in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using NPANXX numbers.

Return Type:

ReverseTaxResults	Contains the calculated base sale amount for the transaction and an array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction	Telecom transaction data

Errors:
- OriginationNpaNxx must be set! – The OriginationNpaNxx property of the Transaction was not set.
- TerminationNpaNxx must be set! – The TerminationNpaNxx property of the Transaction was not set.
- BillToNpaNxx must be set! – The BillToNpaNxx property of the Transaction was not set.
- Transaction is null! – No Transaction passed in
- NPANXX not found. – One of the three NpaNxx properties passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
The desired total charge must be placed in the Charge field of the Transaction.

10.13 CalcReverseAdjWithPCode

It is recommended to begin using the newly renamed 'tax inclusive' API, CalcTaxInclusiveAdjWithPCode, as the 'reverse' API will be deprecated in the future.

This method accepts transaction data and performs tax inclusive adjustments in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using PCodes.

Return Type:

ReverseTaxResults	Contains the calculated base sale amount for the transaction and an array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:
- OriginationPCode must be set! – The OriginationPCode property of the Transaction was not set
- TerminationPCode must be set! – The TerminationPCode property of the Transaction was not set
- BillToPCode must be set! – The BillToPCode property of the Transaction was not set
- Adjustment is null! – No Adjustment passed in
- PCode not found. – One of the three PCode values passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
The desired total charge must be placed in the Charge field of the Transaction as a positive number.

Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

Discounts can also be processed by setting the DiscountType field in the adjustment transaction.

10.14 CalcReverseAdjWithFipsCode

*It is recommended to begin using the newly renamed 'tax inclusive' API, CalcTaxInclusiveAdjWithFipsCode, as the 'reverse' API will be deprecated in the future.

This method accepts transaction data and performs tax inclusive adjustments in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using Fips Codes.

Return Type:

ReverseTaxResults   Contains the calculated base sale amount for the transaction and an
                                    array of TaxData objects that contain the information about the taxes
                                    applied.

Parameters:

Transaction   Telecom transaction data

Errors:
- OriginationFipsCode must be set! – The OriginationFipsCode property of the Transaction was not set.
- TerminationFipsCode must be set! – The TerminationFipsCode property of the Transaction was not set.
- BillToFipsCode must be set! – The BillToFipsCode property of the Transaction was not set.
- Adjustment is null! – No Adjustment passed in
- FIPS no found. – One of the three FIPS codes passed in is not in AFC SaaS Pro
- FIPS (to PCode) cross-reference database not open. – Indicates an error occurred on the server

See also Error Messages Common to all Calculation Methods

Remarks:
The desired total charge must be placed in the Charge field of the Transaction as a positive number.

Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

Discounts can also be processed by setting the DiscountType field in the adjustment transaction.

10.15 CalcReverseAdjWithZipAddress

*It is recommended to begin using the newly renamed 'tax inclusive' API, CalcTaxInclusiveAdjWithZipAddress, as the 'reverse' API will be deprecated in the future.

This method accepts transaction data and performs tax inclusive adjustments in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using ZipAddress objects.

Return Type:

ReverseTaxResults	Contains the calculated base sale amount for the transaction and an array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction	Telecom transaction data

Errors:
- OriginationZipAddress must be set! – The OriginationZipAddress property of the Transaction was not set.
- TerminationZipAddress must be set! – The TerminationZipAddress property of the Transaction was not set.
- BillToZipAddress must be set! – The BillToZipAddress property of the Transaction was not set.
- Adjustment is null! – No Adjustment passed in
- County/State/Zip not found. – One of the three ZipAddress values passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
The desired total charge must be placed in the Charge field of the Transaction as a positive number.

Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

Discounts can also be processed by setting the DiscountType field in the adjustment transaction.

10.16 CalcReverseAdjWithNpaNxx

It is recommended to begin using the newly renamed 'tax inclusive' API, CalcTaxInclusiveAdjWithNpaNxx, as the 'reverse' API will be deprecated in the future.

This method accepts transaction data and performs tax inclusive adjustments in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using ZipAddress objects.

Return Type:

ReverseTaxResults   Contains the calculated base sale amount for the transaction and an array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:
* OriginationNpaNxx must be set! – The OriginationNpaNxx property of the Transaction was not set.
* TerminationNpaNxx must be set! – The TerminationNpaNxx property of the Transaction was not set.
* BillToNpaNxx must be set! – The BillToNpaNxx property of the Transaction was not set.
* Adjustment is null! – No Adjustment passed in
* NPANXX not found. – One of the three NpaNxx properties passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
The desired total charge must be placed in the Charge field of the Transaction as a positive number. Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

Discounts can also be processed by setting the DiscountType field in the adjustment transaction.

10.17 CalcTaxInclusiveTaxesWithPCode

This method accepts transaction data and performs a tax inclusive calculation in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using PCodes.

Return Type:

TaxInclusiveTaxResults   Contains the calculated base sale amount for the transaction and an array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:
- OriginationPCode must be set! – The OriginationPCode property of the Transaction was not set
- TerminationPCode must be set! – The TerminationPCode property of the Transaction was not set
- BillToPCode must be set! – The BillToPCode property of the Transaction was not set
- Transaction is null! – No Transaction passed in
- PCode not found. – One of the three PCode values passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:

The desired total charge must be placed in the Charge field of the Transaction.

10.18 CalcTaxInclusiveTaxesWithFipsCode

This method accepts transaction data and performs a tax inclusive calculation in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using Fips Codes.

Return Type:

TaxInclusiveTaxResults   Contains the calculated base sale amount for the transaction and an array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:
- OriginationFipsCode must be set! – The OriginationFipsCode property of the Transaction was not set.
- TerminationFipsCode must be set! – The TerminationFipsCode property of the Transaction was not set.
- BillToFipsCode must be set! – The BillToFipsCode property of the Transaction was not set.
- Transaction is null! – No Transaction passed in
- FIPS no found. – One of the three FIPS codes passed in is not in AFC SaaS Pro
- FIPS (to PCode) cross-reference database not open. – Indicates an error occurred on the server

See also Error Messages Common to all Calculation Methods

Remarks:

The desired total charge must be placed in the Charge field of the Transaction.

10.19 CalcTaxInclusiveTaxesWithZipAddress

This method accepts transaction data and performs a tax inclusive calculation in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using ZipAddress objects.

Return Type:

TaxInclusiveTaxResults   Contains the calculated base sale amount for the transaction and an array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:
- OriginationZipAddress must be set! – The OriginationZipAddress property of the Transaction was not set.
- TerminationZipAddress must be set! – The TerminationZipAddress property of the Transaction was not set.
- BillToZipAddress must be set! – The BillToZipAddress property of the Transaction was not set.
- Transaction is null! – No Transaction passed in
- County/State/Zip not found. – One of the three ZipAddress values passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
The desired total charge must be placed in the Charge field of the Transaction.

10.20 CalcTaxInclusiveTaxesWithNpaNxx

This method accepts transaction data and performs a tax inclusive calculation in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using NPANXX numbers.

Return Type:

TaxInclusiveTaxResults   Contains the calculated base sale amount for the transaction and an array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:
- OriginationNpaNxx must be set! – The OriginationNpaNxx property of the Transaction was not set.
- TerminationNpaNxx must be set! – The TerminationNpaNxx property of the Transaction was not set.
- BillToNpaNxx must be set! – The BillToNpaNxx property of the Transaction was not set.

- Transaction is null! – No Transaction passed in
- NPANXX not found. – One of the three NpaNxx properties passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
The desired total charge must be placed in the Charge field of the Transaction.

10.21 CalcTaxInclusiveAdjWithPCode

This method accepts transaction data and performs tax inclusive adjustments in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using PCodes.
Return Type:

TaxInclusiveTaxResults   Contains the calculated base sale amount for the transaction and an array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:
- OriginationPCode must be set! – The OriginationPCode property of the Transaction was not set
- TerminationPCode must be set! – The TerminationPCode property of the Transaction was not set
- BillToPCode must be set! – The BillToPCode property of the Transaction was not set
- Adjustment is null! – No Adjustment passed in
- PCode not found. – One of the three PCode values passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
The desired total charge must be placed in the Charge field of the Transaction as a positive number.

Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

Discounts can also be processed by setting the DiscountType field in the adjustment transaction.

10.22 CalcTaxInclusiveAdjWithFipsCode

This method accepts transaction data and performs tax inclusive adjustments in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using Fips Codes.

Return Type:

TaxInclusiveTaxResults   Contains the calculated base sale amount for the transaction and an array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data

Errors:
- OriginationFipsCode must be set! – The OriginationFipsCode property of the Transaction was not set.
- TerminationFipsCode must be set! – The TerminationFipsCode property of the Transaction was not set.
- BillToFipsCode must be set! – The BillToFipsCode property of the Transaction was not set.
- Adjustment is null! – No Adjustment passed in
- FIPS no found. – One of the three FIPS codes passed in is not in AFC SaaS Pro
- FIPS (to PCode) cross-reference database not open. – Indicates an error occurred on the server

See also Error Messages Common to all Calculation Methods

Remarks:
The desired total charge must be placed in the Charge field of the Transaction as a positive number.

Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

Discounts can also be processed by setting the DiscountType field in the adjustment transaction.

10.23 CalcTaxInclusiveAdjWithZipAddress

This method accepts transaction data and performs tax inclusive adjustments in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using ZipAddress objects.

Return Type:

TaxInclusiveTaxResults   Contains the calculated base sale amount for the transaction and an array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction	Telecom transaction data

Errors:
- OriginationZipAddress must be set! – The OriginationZipAddress property of the Transaction was not set.
- TerminationZipAddress must be set! – The TerminationZipAddress property of the Transaction was not set.
- BillToZipAddress must be set! – The BillToZipAddress property of the Transaction was not set.
- Adjustment is null! – No Adjustment passed in
- County/State/Zip not found. – One of the three ZipAddress values passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
The desired total charge must be placed in the Charge field of the Transaction as a positive number.

Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

Discounts can also be processed by setting the DiscountType field in the adjustment transaction.

10.24 CalcTaxInclusiveAdjWithNpaNxx

This method accepts transaction data and performs tax inclusive adjustments in order to arrive at the base sale amount and taxes for the desired total charge. Origination, Termination, and Bill To information is passed using ZipAddress objects.

Return Type:
TaxInclusiveTaxResults	Contains the calculated base sale amount for the transaction and an array of TaxData objects that contain the information about the taxes applied.

Parameters:
Transaction	Telecom transaction data

Errors:
- OriginationNpaNxx must be set! – The OriginationNpaNxx property of the Transaction was not set.
- TerminationNpaNxx must be set! – The TerminationNpaNxx property of the Transaction was not set.
- BillToNpaNxx must be set! – The BillToNpaNxx property of the Transaction was not set.
- Adjustment is null! – No Adjustment passed in
- NPANXX not found. – One of the three NpaNxx properties passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
The desired total charge must be placed in the Charge field of the Transaction as a positive number. Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

Discounts can also be processed by setting the DiscountType field in the adjustment transaction.

10.25 BeginCustomerBatch

Flags the beginning of a customer transaction batch and returns a customer batch ID that can be used to submit each transaction and process the entire batch.

Return Type:

long   A unique identifier for the customer batch.

Parameters:
None

10.26 CalcCustTaxes

Submits a single transaction to be processed for a customer batch.

Return Type:
bool   Boolean indicating success or failure

Parameters:
Long   *Customer batch identifier*
Transaction   Telecom transaction data

Errors:
- Origination location not specified. – The origination location of the Transaction was not set.
- Termination location not specified. – The termination location of the Transaction was not set.
- Bill-To location not specified. – The bill-to location of the Transaction was not set.
- Transaction is null – No transaction passed in.
- Invalid customer batch identifier.
- Maximum number of customer transaction exceeded.

See also Error Messages Common to all Calculation Methods

Remarks:

Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

Discounts can also be processed by setting the DiscountType field in the adjustment transaction.

10.27 CalcCustAdj

Submits a single adjustment to be processed for a customer batch.

Return Type:
bool   Boolean indicating success or failure

Parameters:
Long   *Customer batch identifier*
Transaction   Telecom transaction data

Errors:
- Origination location not specified. – The origination location of the Transaction was not set.
- Termination location not specified. – The termination location of the Transaction was not set.
- Bill-To location not specified. – The bill-to location of the Transaction was not set.
- Adjustment is null – No adjustment passed in.
- Invalid customer batch identifier.
- Maximum number of customer transaction exceeded.

See also Error Messages Common to all Calculation Methods

Remarks:
Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

Discounts can also be processed by setting the DiscountType field in the adjustment transaction.

10.28 ProcessCustomerBatch

Processes a customer batch and returns the tax calculation results.

Return Type:
CustomerResults   Data structure containing an array of TaxData objects for each individual
                                      transaction processed and CustomerTaxData array containing the
                                      summarized taxes for the customer batch.

Parameters:
long   Customer batch ID

bool	Boolean indicating if the detailed taxes for each individual line item should be returned in the results.
Nexus []	An array of Nexus objects. Can be null or an empty list.
Exclusion []	An array of Exclusion objects. Can be null or an empty list.

Errors:
- Invalid customer batch identifier – Customer batch ID is not valid.

See also Error Messages Common to all Calculation Methods

NOTE: This API does not support tax inclusive calls. Please see ProcessCustomerBatchV2 for tax inclusive transactions.

10.29 ProcessCustomerBatchV2

Processes a customer batch and returns the tax calculation results. Individual taxes returned use the TaxDataV2 data structure which contains a TransCharge field for the calculated base sale amount for each corresponding transaction.

Return Type:
| CustomerResultsV2 | Data structure containing an array of TaxDataV2 objects for each individual transaction processed and CustomerTaxData array containing the summarized taxes for the customer batch. |

Parameters:
long	Customer batch ID
bool	Boolean indicating if the detailed taxes for each individual line item should be returned in the results.
Nexus []	An array of Nexus objects. Can be null or an empty list.
Exclusion []	An array of Exclusion objects. Can be null or an empty list.

Errors:
- Invalid customer batch identifier – Customer batch ID is not valid.

See also Error Messages Common to all Calculation Methods

NOTE: Use of this API does incorporate use of tax inclusive transactions.

10.30 CalcTaxesInCustMode

Processes a batch of telecom and/or sales and use transactions and adjustments in Invoice Mode.

Return Type:

CustomerResults   Data structure containing an array of TaxData objects for each individual
                                   transaction processed and CustomerTaxData array containing the
                                   summarized taxes for the customer batch.

Parameters:
Transaction[]   Telecom transaction array
Transaction[]   Telecom transaction array to be processed as adjustments.
SalesUseTransaction[]   Sales and use transaction array to be processed as adjustments (See
                                   TM_00117_AFC SaaS Pro Sales and Use Developer Manual.pdf).
Nexus []   An array of Nexus objects. Can be null or an empty list.
                                   Nexus only applies Sales and Use transactions.
Exclusion []   An array of Exclusion objects. Can be null or an empty list.
bool   Boolean indicating if the detailed taxes for each individual line item
                                   should be returned in the results.

Errors:
See Error Messages Common to all Calculation Methods

NOTE: If a transaction in the batch generates and error, the entire batch will fail. Also, this API does not
support tax inclusive calls. Please see ProcessCustomerBatchV2 for tax inclusive transactions.

10.31 CalcTaxesInCustModeV2

Processes a batch of telecom and/or sales and use transactions and adjustments in Invoice Mode.
Individual taxes returned use the TaxDataV2 data structure which contains a TransCharge field for the
calculated base sale amount for each corresponding transaction.

Return Type:
CustomerResultsV2   Data structure containing an array of TaxDataV2 objects for each
                                   individual transaction processed and CustomerTaxData array containing
                                   the summarized taxes for the customer batch.

Parameters:
Transaction[]   Telecom transaction array
Transaction[]   Telecom transaction array to be processed as adjustments.
SalesUseTransaction[]   Sales and use transaction array to be processed as adjustments (See
                                   TM_00117_AFC SaaS Pro Sales and Use Developer Manual.pdf).
Nexus []   An array of Nexus objects. Can be null or an empty list.
                                   Nexus only applies Sales and Use transactions.
Exclusion []   An array of Exclusion objects. Can be null or an empty list.
SafeHarborOverride[] –   Array of SafeHarborOverride objects. Can be null or an empty
                                   list. These overrides are applied to all Telecom transactions in this
                                   batch. Overrides entered on the individual transactions are ignored.

bool   Boolean indicating if the detailed taxes for each individual line item
                                    should be returned in the results.

Errors:
See Error Messages Common to all Calculation Methods

NOTE: If a transaction in the batch generates and error, the entire batch will fail. Also, use of this API
does incorporate use of tax inclusive transactions.

10.32 CalcProRatedTaxes

Accepts a pro-rated percentage that is used to calculate the taxable amount on the transaction and
perform tax calculations. Bill-to, origination and termination may be entered as PCode, FIPS Codes,
ZipAddress or NPANXX.

Return Type:
TaxData[]   An array of TaxData objects that contain the information about the taxes
                                    applied.

Parameters:
Transaction   Telecom transaction data
Double   Pro-rated percent specified as a decimal.

Errors:
* Origination location not specified. – The origination property of the transaction was not set
* Termination location not specified. – The termination property of the transaction was not set
* Bill-to location not specified.– The bill-to property of the transaction was not set
* Transaction is null! – No Transaction passed in
* PCode not found. – One of the three PCode values passed in is not in AFC SaaS Pro
* FIPS not found. – One of the three FipsCode values passed in is not in AFC SaaS Pro
* County/State/Zip not found. – One of the three ZipAddress values passed in is not in AFC SaaS Pro
* NPANXX not found. – One of the three NpaNxx values passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
None.

10.33 CalcProRatedAdj

Accepts a pro-rated percentage that is used to calculate the taxable amount on the transaction and perform tax adjustments. Bill-to, origination and termination may be entered as PCode, FIPS Codes, ZipAddress or NPANXX.

Return Type:

TaxData[]   An array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction   Telecom transaction data
Double   Pro-rated percent specified as a decimal.

Errors:
- Origination location not specified. – The origination property of the transaction was not set
- Termination location not specified. – The termination property of the transaction was not set
- Bill-to location not specified.– The bill-to property of the transaction was not set
- Transaction is null! – No Transaction passed in
- PCode not found. – One of the three PCode values passed in is not in AFC SaaS Pro
- FIPS not found. – One of the three FipsCode values passed in is not in AFC SaaS Pro
- County/State/Zip not found. – One of the three ZipAddress values passed in is not in AFC SaaS Pro
- NPANXX not found. – One of the three NpaNxx values passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
None.

10.34 CalcJurisdiction

Determines the taxing jurisdiction for a transaction and returns the PCode at the lowest jurisdiction level.

Return Type:

uint   PCode for taxing jurisdiction.

Parameters:

Transaction   Telecom transaction data

Errors:
- Origination location not specified. – The origination property of the transaction was not set
- Termination location not specified. – The termination property of the transaction was not set
- Bill-to location not specified.– The bill-to property of the transaction was not set
- Transaction is null! – No Transaction passed in
- PCode not found. – One of the three PCode values passed in is not in AFC SaaS Pro
- FIPS not found. – One of the three FipsCode values passed in is not in AFC SaaS Pro
- County/State/Zip not found. – One of the three ZipAddress values passed in is not in AFC SaaS Pro

- NPANXX not found. – One of the three NpaNxx values passed in is not in AFC SaaS Pro

See also Error Messages Common to all Calculation Methods

Remarks:
None.

NOTE: The transaction/service pair is required and must be provided in order to determine the correct tax jurisdiction.

Also, please note if CalcJurisdiction is called along with an Exclusion, the Exclusion will not be applied.

10.35 CalcTaxesWithOverrides

Accepts transaction data and tax rate override data to perform tax calculations with provided override information. Bill-to, origination and termination may be entered as PCode, FIPS Codes, ZipAddress or NPANXX.

Return Type:

TaxData[]	An array of TaxData objects that contain the information about the taxes applied.

Parameters:

Transaction	Telecom transaction data
TaxRateOverrideInfo[]	Tax rate override data

Errors:
- Origination location not specified. – The origination property of the transaction was not set
- Termination location not specified. – The termination property of the transaction was not set
- Bill-to location not specified.– The bill-to property of the transaction was not set
- Transaction is null! – No Transaction passed in
- PCode not found. – One of the three PCode values passed in is not in AFC SaaS Pro
- FIPS not found. – One of the three FipsCode values passed in is not in AFC SaaS Pro
- County/State/Zip not found. – One of the three ZipAddress values passed in is not in AFC SaaS Pro
- NPANXX not found. – One of the three NpaNxx values passed in is not in AFC SaaS Pro
- This method cannot be used if you have override file in place. – Override file is in place.
- EZtax failed to insert the override. – Override data cannot be applied.

See also Error Messages Common to all Calculation Methods

Remarks:
This method cannot be used if the user already has an override file in place.

10.36 CalcAdjWithOverrides

Accepts transaction data and tax rate override data to perform tax adjustment calculations with provided override information. Bill-to, origination and termination may be entered as PCode, FIPS Codes, ZipAddress or NPANXX.

Return Type:

| TaxData[] | An array of TaxData objects that contain the information about the taxes applied. |

Parameters:

| Transaction | Telecom transaction data |
| TaxRateOverrideInfo[] | Tax rate override data |

Errors:

* Origination location not specified. – The origination property of the transaction was not set
* Termination location not specified. – The termination property of the transaction was not set
* Bill-to location not specified.– The bill-to property of the transaction was not set
* PCode not found. – One of the three PCode values passed in is not in AFC SaaS Pro
* Transaction is null! – No Transaction passed in
* FIPS not found. – One of the three FipsCode values passed in is not in AFC SaaS Pro
* County/State/Zip not found. – One of the three ZipAddress values passed in is not in AFC SaaS Pro
* NPANXX not found. – One of the three NpaNxx values passed in is not in AFC SaaS Pro
* This method cannot be used if you have override file in place. – Override file is in place.
* EZtax failed to insert the override. – Override data cannot be applied.

See also Error Messages Common to all Calculation Methods

Remarks:
This method cannot be used if the user already has an override file in place.

Adjustments require application of the same override data that is used with CalcTaxesWithOverrides.

Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

Discounts can also be processed by setting the DiscountType field in the adjustment transaction.

10.37 CalcBridgeConferenceTaxes

Note: The Bridge Conferencing feature is currently in development at this time. Please refrain from using this feature as well as the proposed solutions and APIs until further notice.

This method will take a Billing address, Bridge Address, Host Address (Optional) and list of participants and perform the following by participant:

1. Determine whether Interstate or Intrastate taxes apply
2. Determine whether FUSF taxes apply
3. Calculate the taxes
4. Summarize the results

Return Type:

BridgeConferenceResults   Data structure containing an array of BridgeConferenceParticipantResult objects for each participant transaction processed and TaxData array containing the summarized taxes for the bridge conference calculation.

Parameters:

BridgeConferenceTransaction   Bridge conference transaction data structure. Includes an array of BridgeConferenceParticipant objects that define the list of participants.

Errors:
- Bridge conference data is invalid – Invalid transaction data received

See also Error Messages Common to all Calculation Methods

Remarks:
- To return individual participant results, the flag **ReturnParticipantTaxes** must be set to true. Summarized taxes are always returned.
- If **ProcessInvalidParticipant** is set to true, and an invalid participant jurisdiction is encountered, the processing will continue with the greatest tax liability applied for the participant.

10.38 CalcAdjBridgeConferenceTaxes

Note: The Bridge Conferencing feature is currently in development at this time. Please refrain from using this feature as well as the proposed solutions and APIs until further notice.

This method will take a Billing address, Bridge Address, Host Address (Optional) and list of participants and perform the following by participant:

1. Determine whether Interstate or Intrastate taxes apply
2. Determine whether FUSF taxes apply
3. Calculate the adjustment / credit
4. Summarize the results

Return Type:

BridgeConferenceResults

Data structure containing an array of BridgeConferenceParticipantResult objects for each participant transaction processed and TaxData array containing the summarized taxes for the bridge conference calculation.

Parameters:
BridgeConferenceTransaction

Bridge conference transaction data structure. Includes an array of BridgeConferenceParticipant objects that define the list of participants.

Errors:
• Bridge conference data is invalid – Invalid transaction data received

See also Error Messages Common to all Calculation Methods

Remarks:
• To return individual participant results, the flag **ReturnParticipantTaxes** must be set to true. Summarized taxes are always returned.
• If **ProcessInvalidParticipant** is set to true, and an invalid participant jurisdiction is encountered, the processing will continue with the greatest tax liability applied for the participant.
• Adjustment transactions are primarily used to process refunds or credits. They rely on the AdjustmentMethod field.

10.39 CommitTransactions

The CommitTransactions API is used to commit or uncommit a DocumentCode.

Return Type:
bool   Boolean indicating success or failure

Note: Exception thrown with appropriate error message in case of failure.

Parameters:
Transaction   Telecom transaction data
CommitData   Required as a parameter in order to specify a valid DocumentCode and a Boolean indicating whether transactions with the specified DocumentCode should be committed or uncommitted. Any optional field values provided in this parameter are used when reports are generated in order to replace the corresponding values originally specified within the transaction at the time of tax calculation processing.

Errors:
• DocumentCode cannot be blank or null.

- DocumentCode not found.
- DocumentCode has been locked.
- DoccumentCode cannot exceed 150 characters.

See also Error Messages Common to all Calculation Methods

Remarks:
None.

10.40 Error Messages Common to all Calculation Methods

These messages can be returned by any of the calculation methods:
- AdjustmentMethod is Invalid! – The Adjustment Method passed in is not valid
- DiscountType is Invalid! – The Discount Type passed in is not valid
- No Session Available! – Indicates that no sessions were available for your request. You may be making too many simultaneous requests to the web service or there is an issue that needs to be reported.

- Authorization Error – Indicates that your userid or password is not recognized.
- Not Authorized! – Indicates that you have called a method on the web service that your service agreement does not include.
- Server Fault! – Indicates an error occurred on the server.
- CustomerType is Invalid! – The Customer Type passed in is not valid
- BusinessClass is Invalid! – The Business Class passed in is not valid
- ServiceClass is Invalid! – The Service Class passed in is not valid
- Jurisdiction not found. – The jurisdiction does not exist in AFC SaaS Pro
- Invalid Transaction Date – The Transaction Date passed in is not valid
- Invalid transaction/service pair – The Transaction Type and Service Type combination is not valid
- Transaction/Service pair is not supported by the API call – The Transaction Type and Service Type combination cannot be used by the Telecom Interface
- Session not initialized – Indicates an error occurred on the server
- JCode database not open. – Indicates an error occurred on the server
- PCode database not open. – Indicates an error occurred on the server
- ZipCode database not open. – Indicates an error occurred on the server
- NPANXX database not open. – Indicates an error occurred on the server
- Address database not open. – Indicates an error occurred on the server
- Unable to start EZtax. – Indicates an error occurred on the server
- filelocs.txt configuration file not found. – Indicates an error occurred on the server
- Unable to read filelocs.txt configuration file – Indicates an error occurred on the server
- An EZtaxSession hasn't been specified for this transaction – Indicates an error occurred on the server
- The EZtaxSession object has been disposed. – Indicates an error occurred on the server
- Unable to return the log information – Indicates an error occurred on the server
- Split for private line transaction must be greater than or equal to 0 and less than or equal to 1.
- A valid Transaction Type or Service Type must be supplied.
- Supplied Service Type is invalid.
- Supplied Transaction Type is invalid.
- Cross country border transaction does not support transaction/service auto determination.
- Participants cannot be mixed jurisdiction types!
- PCode not found.
- Address not found.

11.SOAP Utility Web Methods

These methods provide a variety of information from the AFC SaaS Pro Web Service.

Method Name	Summary
GetAddress	Returns an array of addresses that correspond to a PCode
GetTaxCategory	Returns the tax category for a tax type

Method Name	Summary
GetTaxDescription	Returns the tax description for a tax type
GetTaxRates	Returns the tax rate information for a jurisdiction
FipsToPCode	Returns the PCode that corresponds to a FIPS code
PCodeToFips	Returns the FIPS code that corresponds to a PCode
ZipToPCode	Returns the PCode that corresponds to a Zip Address
NpaNxxToPCode	Returns the PCode that corresponds to an NpaNxx value
GetServerTime	Returns the server time on the AFC SaaS Pro Web Service
GetVersion	Returns the version of the AFC SaaS Pro Web Service
GetEZtaxVersion	Returns the version of the underlying AFC engine
GetEZtaxDbVersion	Returns the version of the database used by the underlying AFC engine
CreateReport	Submits a request for generating a report based on transactions and taxes that have been processed in the web service.
GetOptionalFieldKeyDesc	Returns the description for each optional field key in the OptionalField array of the Transaction. The values specified will be used when generating reports that contain these optional fields.
UpdateOptionalFieldKeyDesc	Updates the description for each optional field key thereby indicating what each field in the OptionalField array of the Transaction is used for (e.g. invoice number, line item number, etc.). The values specified will be used when generating reports that contain these optional fields.
ZipLookup	Returns all or multiple PCodes and juridisdiction details associated with the address input provided by clients.

11.1 GetAddress

This method returns the addresses for the specified jurisdiction. If the jurisdiction is invalid the return will be NULL.

Return Type:

AddressData[]   An array of AddressData objects that contain the addresses for the jurisdiction specified by the supplied PCode.

Parameters:

uint   The PCode for the desired jurisdiction.

Errors:
- No Session Available! – Indicates that no sessions were available for your request. You may be making too many simultaneous requests to the web service or there is an issue that needs to be reported.
- Authorization Error – Indicates that your userid or password is not recognized.
- Not Authorized! – Indicates that you have called a method on the web service that your service agreement does not include.
- Server Fault! – Indicates an error occurred on the server.
- PCode not found.

Remarks:
Some jurisdictions have multiple valid addresses.

11.2 GetTaxCategory

This method returns the tax category for the specified tax type. If the tax type is invalid the return will be NULL or an empty string.

Return Type:

string   A string describing the tax category for the specified tax type.

Parameters:

int   The tax type.

Errors:
- No Session Available! – Indicates that no sessions were available for your request. You may be making too many simultaneous requests to the web service or there is an issue that needs to be reported.
- Authorization Error – Indicates that your userid or password is not recognized.
- Not Authorized! – Indicates that you have called a method on the web service that your service agreement does not include.
- Server Fault! – Indicates an error occurred on the server.

Remarks:
None.

11.3 GetTaxDescription

This method returns the description for the specified tax type. If the tax type is invalid the return will be NULL or an empty string.

Return Type:

string   A string describing the specified tax type.

Parameters:

int   The tax type.

Errors:
- No Session Available! – Indicates that no sessions were available for your request. You may be making too many simultaneous requests to the web service or there is an issue that needs to be reported.
- Authorization Error – Indicates that your userid or password is not recognized.

- Not Authorized! – Indicates that you have called a method on the web service that your service agreement does not include.
- Server Fault! – Indicates an error occurred on the server.

Remarks:
None.

11.4 GetTaxRates

Determines the tax rate information for a jurisdiction identified by the input PCode.

Return Type:
TaxRateInfo   Tax Rate information for a jurisdiction.

Parameters:
uint   PCode for taxing jurisdiction

Errors:
- PCode not found. – The PCode value passed in is not in AFC SaaS Pro.

See also Error Messages Common to all Calculation Methods

Remarks:
None.

11.5 FipsToPCode

This method returns the PCode for the specified FIPS code. If the FIPS Code is invalid or has no AFC jurisdiction the return will be NULL.

Return Type:
uint   A nullable unsigned int for the PCode.

Parameters:
string   The FIPS code.

Errors:
- No Session Available! – Indicates that no sessions were available for your request. You may be making too many simultaneous requests to the web service or there is an issue that needs to be reported.

- Authorization Error – Indicates that your userid or password is not recognized.
- Not Authorized! – Indicates that you have called a method on the web service that your service agreement does not include.

Remarks:
None.

11.6   PCodeToFips

This method returns the FIPS Code for the specified PCode. If the PCode Code is invalid the return will be NULL.

Return Type:

string   A string for the FIPS Code.

Parameters:

uint   The PCode.

Errors:
- No Session Available! – Indicates that no sessions were available for your request. You may be making too many simultaneous requests to the web service or there is an issue that needs to be reported.
- Authorization Error – Indicates that your userid or password is not recognized.
- Not Authorized! – Indicates that you have called a method on the web service that your service agreement does not include.
- Server Fault! – Indicates an error occurred on the server.

Remarks:
None.

11.7   ZipToPCode

This method returns the PCode for the specified ZIP Address. If the ZIP address is invalid or has no AFC jurisdiction the return will be NULL.

Return Type:

uint   A nullable unsigned int for the PCode. Null indicates an invalid address.

Parameters:

ZipAddress   The ZipAddress object.

Errors:

- No Session Available! – Indicates that no sessions were available for your request. You may be making too many simultaneous requests to the web service or there is an issue that needs to be reported.
- Authorization Error – Indicates that your userid or password is not recognized.
- Not Authorized! – Indicates that you have called a method on the web service that your service agreement does not include.
- Server Fault! – Indicates an error occurred on the server.

Remarks:
Based upon the best match, only one PCode is returned. To obtain a full list of matching PCodes, please reference the [ZipLookup](#) API. Also, please note it is not required to provide the State. Please reference [Zip Lookup Requests](#) for additional information and details.

11.8 NpaNxxToPCode

This method returns the PCode for the specified NpaNxx value. If the NpaNxx value is invalid or has no AFC jurisdiction the return will be NULL.

Return Type:
uint   A nullable unsigned int for the PCode

Parameters:
uint   The NpaNxx value.

Errors:
- No Session Available! – Indicates that no sessions were available for your request. You may be making too many simultaneous requests to the web service or there is an issue that needs to be reported.
- Authorization Error – Indicates that your userid or password is not recognized.
- Not Authorized! – Indicates that you have called a method on the web service that your service agreement does not include.
- Server Fault! – Indicates an error occurred on the server.
- PCode not found.

Remarks:
None.

11.9 GetServerTime

This method returns the time on the AFC SaaS Pro Web Service.

Return Type:
DateTime   The AFC SaaS Pro Web Service server time.

Parameters:
None

Errors:
- No Session Available! – Indicates that no sessions were available for your request. You may be making too many simultaneous requests to the web service or there is an issue that needs to be reported.
- Authorization Error – Indicates that your userid or password is not recognized.
- Not Authorized! – Indicates that you have called a method on the web service that your service agreement does not include.
- Server Fault! – Indicates an error occurred on the server.

Remarks:
None.

11.10 GetVersion

This method returns the time on the AFC SaaS Pro Web Service.

Return Type:
string   The version of the AFC SaaS Pro Web Service.

Parameters:
None

Errors:
- Authorization Error – Indicates that your userid or password is not recognized.
- Server Fault! – Indicates an error occurred on the server.

Remarks:
None.

11.11 GetEZtaxVersion

This method returns the version of the AFC engine used in the background for performing tax calculations.

Return Type:
string   The version of the AFC engine.
Parameters:
None

Errors:
- Authorization Error – Indicates that your userid or password is not recognized.

- Server Fault! – Indicates an error occurred on the server.

Remarks:
None.

11.12 GetEZtaxDbVersion

This method returns the version of the underlying AFC database used by the AFC engine.

Return Type:

string   The version of the AFC database.

Parameters:
None
Errors:
- Authorization Error – Indicates that your userid or password is not recognized.
- Server Fault! – Indicates an error occurred on the server.

Remarks:
None.

11.13 CreateReport

Submits a request for generating a custom report based on transactions and taxes that have been processed in the web service.

Return Type:

boolean   Boolean indicating success or failure.

Parameters:

ReportOptions   Data structure containing the options for the report.

Errors:
- reportOptions parameter must be specified.
- ReportType must be specified.
- StartDate for the report may not be a future date.
- StartDate for the report may not be prior than 91 days.
- TimeSpan cannot be greater than 31 days.
- TimeSpan or EndDate must be specified.
- EndDate cannot be more than 31 days apart from StartDate.
- EndDate must be greater than StartDate.
- Precision must be between 0 and 6.

- FileName may not exceed 100 characters
- FileExtension may not exceed 5 characters.
- EmailAddress cannot be longer than 100 characters.
- EmailAddress is not in a valid format.
- CustomLogFields must be specified if BaseReport is not used.

Remarks:
None.

Note: Only transactions and taxes that include the company's 3-character company code in the CompanyIdentifier field of the transaction will be included in the report.

11.14 GetOptionalFieldKeyDesc

Returns the description for each optional field key in the OptionalField array of the Transaction. The values specified will be used when generating reports that contain these optional fields.

Return Type:
OptionalKey[]   Array containing key numbers and descriptions for each key. The OptionalKeyDesc field of the OptionalKey object contains the description for the OptionalKeyNo.

Parameters:
None

Errors:
- Authorization Error – Indicates that your userid or password is not recognized.
- Server Fault! – Indicates an error occurred on the server.

Remarks:
None.

11.15 UpdateOptionalFieldKeyDesc

Updates the description for each optional field key thereby indicating what each field in the OptionalField array of the Transaction is used for (e.g. invoice number, line item number, etc.). The values specified will be used when generating reports that contain these optional fields.

Return Type:
Boolean   Indicates success or failure.

Parameters:

OptionalKey[]   Array containing key numbers and descriptions for each key. The
                                       OptionalkeyDesc field of the OptionalKey object contains the description
                                       for the OptionalKeyNo.

Errors:
- Authorization Error – Indicates that your userid or password is not recognized.
- Server Fault! – Indicates an error occurred on the server.

Remarks:
None.

11.16 ZipLookup

Returns all or multiple PCodes and juridisdiction details associated with the address input provided by clients.

Return Type:
ZipLookupResult   The ZipLookupResult object.

Parameters:
ZipLookup   The ZipLookup object.

Errors:
- No Session Available! – Indicates that no sessions were available for your request. You may be making too many simultaneous requests to the web service or there is an issue that needs to be reported.
- Authorization Error – Indicates that your userid or password is not recognized.
- Not Authorized! – Indicates that you have called a method on the web service that your service agreement does not include.
- Server Fault! – Indicates an error occurred on the server.
- No jurisdiction data set – Indicates the request had no valid jurisdiction data set (at a minimum, one jurisdiction field must be set).
- Address not found.

Remarks:
If Country is not specified, the default value is set to "USA".

12.REST Interface APIs

The APIs are also available via REST. The REST URL is https://communications.avalara.net. To access a specific REST API, access the API via the URLs listed in sections 9 and 10. The table below provides an overview as well.

Example: https://communications.avalara.net/api/v1/Application/ServerTime

In order to use the REST APIs, an api_key needs to be passed as authentication. The api_key is a Base64-encoded value is the form of "UserId:Password" where the UserId and Password are the email address and password used to log into the AvaTax for Communications Customer Portal. To encode the api_key, use a site such as https://www.base64encode.org/.

Users with access to multiple clients are required to provide the numerical Client ID for the specific client on behalf of whom they are making the call. To authenticate, use the Base64 encoded header value for api_key in the form of "UserId:Password" along with header values for client_id and client_profile_id. (For additional details on Client IDs and Profile IDs, please reference Use of Client IDs in the REST Interface.

Example: The UserId is email@sample.com and the Password is Test123. The initial Base64 decoded value is email@sample.com:Test123. The resulting Base64 encoded api_key is ZW1haWxAc2FtcGxlLmNvbTpUZXN0MTIz.

Note: The Client_ID is the Client ID provided by Avalara. Also, Client_profile_id is currently used and must be populated with a value if a specific profile is desired. If left blank, the default profile ID of zero will be used.

A table providing an overview of the corresponding REST APIs is provided below. For further information regarding the REST APIs, please see https://developer.avalara.com/api-reference/communications/afc-rest/.

SaaS Pro - REST	SaaS Pro – SOAP	REST Description
CalculateTaxes	CalcTaxesWithPCode	Calculate taxes on a transaction.
	CalcTaxesWithNpaNxx	
	CalcTaxesWithZipAddress	
	CalcTaxesWithFipsCode	
CalculateAdjustments	CalcAdjWithPCode	Calculate taxes adjustments on a transaction.
	CalcAdjWithNpaNxx	
	CalcAdjWithZipAddress	
	CalcAdjWithFipsCode	
CalculateTaxInclusive/Taxes	CalcReverseTaxesWithPCode CalcTaxInclusiveTaxesWithPCode	Perform a tax inclusive calculation.
	CalcReverseTaxesWithNpaNxx CalcTaxInclusiveTaxesWithNpaNxx	
	CalcReverseTaxesWithZipAddress CalcTaxInclusiveTaxesWithZipAddress	
	CalcReverseTaxesWithFipsCode CalcTaxInclusiveTaxesWithFipsCode	
CalculateTaxInclusive/Adjustments	CalcReverseAdjWithPCode CalcTaxInclusiveAdjWithPCode	Perform a tax inclusive adjustment.
	CalcReverseAdjWithNpaNxx CalcTaxInclusiveAdjWithNpaNxx	
	CalcReverseAdjWithZipAddress CalcTaxInclusiveAdjWithZipAddress	
	CalcReverseAdjWithWithFipsCode CalcTaxInclusiveAdjWithFipsCode	

SaaS Pro - REST	SaaS Pro – SOAP	REST Description
CalculateProRated/Taxes	CalcProRatedTaxes	Calculate taxes for a pro-rated transaction.
CalculateProRated/Adjustments	CalcProRatedAdj	Calculate tax adjustments for a pro-rated transaction.
CustMode/BeginBatch	BeginCustomerBatch	Initiate a new customer batch.
CustMode/ProcessBatch	ProcessCustomerBatch	Process a customer batch.
CustMode/CalcTaxes	CalcCustTaxes	Submit a transaction for a customer batch.
CustMode/CalcAdjustments	CalcCustAdj	Submit an adjustment for a customer batch.
CustMode/CalCTaxesInCustMode	CalcTaxesInCustMode	Process transactions and adjustments in Invoice Mode.
CalculateJurisdiction	CalcJurisdiction	Determine jurisdiction for transaction.
CalculateWithOverrides/Taxes	CalcTaxesWithOverrides	Calculate taxes using overrides.
CalculateWithOverrides/Adjustments	CalcAdjWithOverrides	Calculate tax adjustments using overrides.
CommitTransactions	CommitTransactions	Calculate taxes on a transaction.
BridgeConference/Taxes	CalcBridgeConferenceTaxes	Calculate taxes on a bridge conference transaction.
BridgeConference/Adjustments	CalcAdjBridgeConferenceTaxes	Calculate tax adjustments on a bridge conference transaction.
Location/PCode	FipsToPCode	Determine PCode for a location.
	ZipToPCode	
	NpaNxxToPCode	
Location/PCodeToFips/{pCode}	PCodeToFips	Convert a PCode to FIPS code.
Location/Address/{pCode}	Get Address	Get address information for a jurisdiction.
Location/ZipAddressLookup	ZipLookup	Lookup jurisdictions by location name and/or postal code.
TaxLookup/Category/{taxType}	GetTaxCategory	Get tax category for a tax type.
TaxLookup/Description/{taxType}	GetTaxDescription	Get the tax description for a tax type.
TaxLookup/TaxRates/{pCode}	GetTaxRates	Get tax rates for a jurisdiction.
Application/ServerTime	GetServerTime	Get server time.
Application/AFCEngineVersion	GetEZtaxVersion	Get the version of the AFC tax engine.
Application/RESTVersion	GetVersion	Get the version of the Rest APIs.
Application/AFCDatabaseEngineVersion	GetEZtaxDbVersion	Get the AFC tax engine database version.
TaxLookup/TransactionServices	N/A	Get list of all Transaction-Service pairs including charge, lines or minutes as a required field.
Location/Address	N/A	Lookup all countries in AFC
Location/Address/{countryIso}	N/A	Lookup all states in a given country.

SaaS Pro - REST	SaaS Pro – SOAP	REST Description
Location/Address/{countryIso}/{stateCode}	N/A	Lookup all cities for a given country and state/province/region.
CalculateTaxesMulti	N/A	Calculate multi transactions.

13. Web Service Data Definitions

Transaction

Data Definition

Property Name	Type	Description
CustomerType	int	One of the CustomerType enums
BusinessClass	int	One of the BusinessClass enums
Sale	bool	true = Sale, false = Resale (Wholesale)
TransactionType	short	See the Transaction Mapping guidelines for valid transaction/service pairs
ServiceType	short	
ServiceClass	int?	One of the ServiceClass enums
Date	DateTime	Date of transaction
Charge	double	Amount to tax
Incorporated	bool	true = address inside city limits false = unincorporated address
FederalExempt	bool	true = exempt federal level taxes
StateExempt	bool	true = exempt state level taxes
CountyExempt	bool	true = exempt county level taxes
LocalExempt	bool	true = exempt city/local taxes
FederalPCode	uint	federal PCode to exempt
StatePCode	uint	state PCode to exempt
CountyPCode	uint	county PCode to exempt
LocalPCode	uint	local PCode to exempt
Exclusions	Exclusion[]	Array of Exclusion objects to apply to this transaction
Exemptions	TaxExemption[]	Array of TaxExemption objects to apply to this transaction
CategoryExemptions	CategoryExemption[]	Array of CategoryExemption objects to apply this transaction
ExemptionType	int	set to 0 - reserved for future use
InvoiceNumber	uint	optional invoice or order number
Optional	uint	optional field
CustomerNumber	string	optional customer account number (up to 20 characters)
CompanyIdentifier	string	optional company identifier (up to 20 characters)
OptionalAlpha1	string	optional field up to 20 characters
Optional4	uint	optional field

	Property Name	Type	Description
Transaction			
Data Definition			
	Optional5	uint	optional field
	Optional6	uint	optional field
	Optional7	uint	optional field
	Optional8	uint	optional field
	Optional9	uint	optional field
	Optional10	uint	For live transactions, this field must indicate the billing cycle in YYYYMM format.
	AdjustmentMethod	int	for adjustments only (otherwise zero) One of the AdjustmentMethod enums
	OriginationAddress	ZipAddress	ZipAddress object for origination – required if using ZIP address calculations otherwise NULL
	OriginationFipsCode	string	FIPS Code for origination – required if FIPS calculation, otherwise can be NULL
	OriginationPCode	uint?	PCode for origination – required if PCode calculation, otherwise can be NULL
	OriginationNpaNxx	uint?	NpaNxx for origination – required if NpaNxx calculation, otherwise can be NULL
	TerminationAddress	ZipAddress	ZipAddress object for termination – required if using ZIP address calculations otherwise NULL
	TerminationFipsCode	string	FIPS Code for termination – required if FIPS calculation, otherwise can be NULL
	TerminationPCode	uint?	PCode for termination – required if PCode calculation, otherwise can be NULL
	TerminationNpaNxx	uint?	NpaNxx for termination – required if NpaNxx calculation, otherwise can be NULL
	BillToAddress	ZipAddress	ZipAddress object for Bill To – required if using ZIP address calculations otherwise NULL
	BillToFipsCode	string	FIPS Code for Bill To – required if FIPS calculation, otherwise can be NULL
	BillToPCode	uint?	PCode for Bill To – required if PCode calculation, otherwise can be NULL
	BillToNpaNxx	uint?	NpaNxx for Bill To – required if NpaNxx calculation, otherwise can be NULL
	Lines	int	For tax calculations based on line counts, it is the number of lines used in the calculation to produce the tax. For Federal taxes, the line count always matches the lines input. For all other taxes, it is 0.
	Locations	int	Number of customer locations
	Minutes	double	For tax calculations based on minutes, it is the number of minutes used in the calculation to produce the tax. For all other taxes, it is 0.
	Debit	bool	true = debit (prepaid)
	DiscountType	int	One of the DiscountType enums
	FacilitiesBased	bool	true = facilities based
	Franchise	bool	true = franchise
	Lifeline	bool	true = lifeline customer

Transaction

Data Definition

	Property Name	Type	Description
	Regulated	bool	true = regulated
	ServiceLevelNumber	uint	Optional field
	OptionalFields	OptionalField[]	Array of optional alphanumeric values to associate with the transaction. Each entry in the array must have an appropriate key number from 1 to 10 indicating to which optional field key the value corresponds to.
	TaxInclusive	Bool	For use with Invoice Mode APIs only. Indicates if the charge amount of the transaction includes the tax amount.
	SafeHarborOverrides	SafeHarborOverride[]	Array of SafeHarborOverride objects to apply to this transaction.
	IsPrivateLine PrivateLineSplit		True/False; Percentage allocation specified as a decimal in the range 0 to 1.

ZipAddress

Data Definition

	Property Name	Type	Description
	CountryISO	string	3 character ISO abbreviation
	County	string	County name
	Locality	string	City name
	State	string	State Abbreviation
	ZipCode	string	ZIP Code (5 digit)
	ZipP4	string	ZIP Code extension (4 digit)

AddressData

Data Definition

	Property Name	Type	Description
	CountryISO	string	3 character ISO abbreviation
	County	string	County name
	Locality	string	City name
	State	string	State Abbreviation
	TaxLevel	int	one of the TaxLevel enums
	ZipBegin	string	First 5-digit ZIP Code in range
	ZipEnd	string	Last 5-digit ZIP Code in range
	ZipP4Begin	string	First 4-digit ZIP Code extension in range
	ZipP4End	string	Last 4-digit ZIP Code extension in range

Nexus

Data Definition

	Property Name	Type	Description
	CountryISO	string	3 character ISO abbreviation
	HasNexus	bool	true = has nexus in this state

Nexus			
Data Definition			
	Property Name	Type	Description
	Locality	string	State Abbreviation

Exclusion			
Data Definition			
	Property Name	Type	Description
	CountryISO	string	3 character ISO abbreviation
	ExclusionOn	bool	true = exclude specified Country/State
	State	string	State Abbreviation

TaxExemption			
Data Definition			
	Property Name	Type	Description
	PCode	uint	PCode for jurisdiction to exempt
	TaxLevel	int	One of the TaxLevel enums (less than 5)
	TaxType	short	Valid AFC tax type

TaxData			
Data Definition			
	Property Name	Type	Description
	AdjustmentType	int	0 if not an adjustment, otherwise one of the DiscountType enums based on user input
	Billable	bool	true = billable
	CalculationType	Int	one of the CalculationType enums
	CategoryDescription	string	tax category description
	CategoryID	short	tax category id
	CompanyIdentifier	string	matches user input in transaction data
	Compliance	bool	true = reportable to jurisdiction
	CustomerNumber	string	matches user input in transaction data
	Description	string	tax description
	ExemptionType	int	set to 0 - reserved for future use
	ExemptSaleAmount	double	amount of sale not subject to tax
	InvoiceNumber	uint	matches user input in transaction data
	Lines	int	For tax calculations based on line counts, it is the number of lines used in the calculation to produce the tax. For Federal taxes, the line count always matches the lines input. For all other taxes, it is 0.
	Locations	int	number of locations in transaction data
	Minutes	int	For tax calculations based on minutes, it is the number of minutes used in the calculation to produce the tax. For all other taxes, it is 0.
	Optional	uint	matches user input in transaction data
	Optional4	uint	matches user input in transaction data

TaxData		
Data Definition		
Property Name	Type	Description
Optional5	uint	matches user input in transaction data
Optional6	uint	matches user input in transaction data
Optional7	uint	matches user input in transaction data
Optional8	uint	matches user input in transaction data
Optional9	uint	matches user input in transaction data
Optional10	uint	matches user input in transaction data
OptionalAlpha1	string	matches user input in transaction data
PCode	uint	PCode for the reporting jurisdiction
Rate	double	Tax rate
RefundUncollect	double	amount of sale refunded or written off
ServiceLevelNumber	uint	matches user input in transaction data
Surcharge	bool	true = tax considered a surcharge
TaxableMeasure	double	amount of sale subject to tax
TaxAmount	double	tax amount
TaxLevel	int	one of the TaxLevel enums
TaxType	short	one of the Avalara tax types

TaxDataV2		
Data Definition		
Property Name	Type	Description
AdjustmentType	int	0 if not an adjustment, otherwise one of the DiscountType enums based on user input
Billable	bool	true = billable
CalculationType	Int	one of the CalculationType enums
CategoryDescription	string	tax category description
CategoryID	short	tax category id
CompanyIdentifier	string	matches user input in transaction data
Compliance	bool	true = reportable to jurisdiction
CustomerNumber	string	matches user input in transaction data
Description	string	tax description
ExemptionType	int	set to 0 - reserved for future use
ExemptSaleAmount	double	amount of sale not subject to tax
InvoiceNumber	uint	matches user input in transaction data
Lines	int	For tax calculations based on line counts, it is the number of lines used in the calculation to produce the tax. For Federal taxes, the line count always matches the lines input. For all other taxes, it is 0.
Locations	int	number of locations in transaction data
Minutes	int	For tax calculations based on minutes, it is the number of minutes used in the calculation to produce the tax. For all other taxes, it is 0.
Optional	uint	matches user input in transaction data
Optional4	uint	matches user input in transaction data
Optional5	uint	matches user input in transaction data
Optional6	uint	matches user input in transaction data

TaxDataV2

	Property Name	Type	Description
	Optional7	uint	matches user input in transaction data
	Optional8	uint	matches user input in transaction data
	Optional9	uint	matches user input in transaction data
	Optional10	uint	matches user input in transaction data
	OptionalAlpha1	string	matches user input in transaction data
	PCode	uint	PCode for the reporting jurisdiction
	Rate	double	Tax rate
	RefundUncollect	double	amount of sale refunded or written off
	ServiceLevelNumber	uint	matches user input in transaction data
	Surcharge	bool	true = tax considered a surcharge
	TaxableMeasure	double	amount of sale subject to tax
	TaxAmount	double	tax amount
	TaxLevel	int	one of the TaxLevel enums
	TaxType	short	one of the Avalara tax types
	TransCharge	double	For tax inclusive transactions, this is the base sale amount calculated by AFC SaaS Pro required in order to arrive at the total desired charge.

Data Definition

CustomerResults

Data Definition

	Property Name	Type	Description
	Taxes	TaxData[]	Data structure containing an array of TaxData objects for each individual transaction processed.
	SummarizedTaxes	CustomerTaxData[]	Data structure containing an array of CustomerTaxData objects with summarized taxes for all transactions processed.

CustomerResultsV2

Data Definition

	Property Name	Type	Description
	Taxes	TaxDataV2[]	Data structure containing an array of TaxData objects for each individual transaction processed.
	SummarizedTaxes	CustomerTaxData[]	Data structure containing an array of CustomerTaxData objects with summarized taxes for all transactions processed.

CustomerTaxData

	Property Name	Type	Description
	PCode	uint	PCode for the reporting jurisdiction
	TaxType	int	one of the Avalara tax types
	TaxLevel	int	one of the TaxLevel enums
	CalculationType	int	enum
	Rate	double	one of the CalculationType enums
	TaxAmount	double	tax amount
	ExemptSaleAmount	double	amount of sale not subject to tax
	Lines	int	For tax calculations based on line counts, it is the number of lines used in the calculation to produce the tax. For Federal taxes, the line count always matches the lines input. For all other taxes, it is 0.
	Locations	int	sum of locations from customer input
	Minutes	int	For tax calculations based on minutes, it is the number of minutes used in the calculation to produce the tax. For all other taxes, it is 0.
	Description	string	tax description
	Surcharge	short	true = tax considered a surcharge
	MaxBase	double	max amount to which tax is applied
	MinBase	double	min amount to which tax is applied
	ExcessTax	double	rate for amount above max base
	TotalCharge	double	sum of charges for this customer
	CategoryID	short	tax category id
	CategoryDescription	string	tax category description

Data Definition

ReverseTaxResults

Data Definition

	Property Name	Type	Description
	BaseSaleAmount	double	Base sale amount calculated by AFC SaaS Pro. This is the charge amount required in order to arrive at the total desired charge.
	Taxes	TaxData[]	Taxes returned for the transaction.

TaxInclusiveTaxResults

Data Definition

	Property Name	Type	Description
	BaseSaleAmount	double	Base sale amount calculated by AFC SaaS Pro. This is the charge amount required in order to arrive at the total desired charge.
	Taxes	TaxData[]	Taxes returned for the transaction.

TaxRateInfo

Data Definition		
Property Name	Type	Description
TaxLevel	Short	Tax Level number
TaxType	Short	Tax Type number
RateHistory	List of TaxRateHistory	Tax Rate History

TaxRateHistory		
Data Definition		
Property Name	Type	Description
EffectiveDate	DateTime	Effect date of this tax rate
LevelExemptible	Boolean	True or false indicate if it is level exemptible
BracketInfo	List of TaxBracketInfo	List of tax Brackets

TaxBracketInfo		
Data Definition		
Property Name	Type	Description
CountyOverrideOn	Boolean	Indicate county override
CountyOverrideTax	Double	County override tax rate
StateOverrideOn	Boolean	Indicate State override
StateOverrideTax	Double	State Override tax rate
Rate	Double	Tax Rate normally 0 to less than 1
MaxBase	Double	The cap for this tax Rate

ReportOptions		
Data Definition		
Property Name	Type	Description
StartDate	DateTime	Beginning of date range for data to be included in the report.
EndDate	DateTime	End of date range for data to be included in the report. This field is not required if TimeSpan is specified.
TimeSpan	TimeSpan	TimeSpan for the date range to be included in the report (e.g. number of days after StartDate). This field is not required if EndDate is specified.
ReportType	String	Name of report to be generated.
IncludeHeaders	Boolean	Indicates if the columns headers should be included in the first row of the report output.
CreateNbaFile	Boolean	Indicates if non-billable amounts should be placed in a separate file.
CreateNcaFile	Boolean	Indicates if non-compliance amounts should be placed in a separate file.
FileName	String	Name for output file (up to 100 characters).
FileExtension	String	Extension for output file (up to 5 characters).
EmailAddress	String	Email address for receiving an email notification when the report has been generated (up to 100 characters).
Precision	Int	Number of decimal places for computed values.

ReportOptions

Data Definition			
	Property Name	**Type**	**Description**
	BaseReport	String	When generating a "customlog" report, indicates which report type to use as a starting template.
	CustomLogFields	List of CustomLogField	List of columns to include in the report when using the "customlog" report type.

CustomLogField

Data Definition			
	Property Name	**Type**	**Description**
	Column	String	Name of column. Refer to 4.4 – Custom Log Report Columns.
	Include	Boolean	Indicates if the column should be included in the output.
	Sort	Boolean	Indicates if the column should be used for sorting.

TaxRateOverrideInfo

Data Definition			
	Property Name	**Type**	**Description**
	PCode	uint	PCode for the reporting jurisdiction
	Scope	Short	Override Scope
	TaxLevel	Short	Tax Level number
	TaxType	Short	Tax Type number
	LevelExemptible	Boolean	Make tax type level exemptible or not
	BracketInfo	List of TaxBracketInfo	List of Tax Brackets

CategoryExemption

Data Definition			
	Property Name	**Type**	**Description**
	CountryISO	String	3 character ISO abbreviation
	State	String	State Abbreviation
	TaxCategory	short	Valid tax category ID

TaxLogDataV914

Data Definition			
	Property Name	**Type**	**Description**
	AdjustmentType	int	0 if not an adjustment, otherwise one of the DiscountType enums based on user input
	Billable	bool	true = billable
	CalcType	Int	one of the CalculationType enums
	CategoryDescription	string	tax category description
	CategoryID	short	tax category id
	CompanyIdentifier	string	matches user input in transaction data

TaxLogDataV914		
Data Definition		
Property Name	**Type**	**Description**
Compliance	bool	true = reportable to jurisdiction
CustomerNumber	string	matches user input in transaction data
Description	string	tax description
ExemptSaleAmount	double	amount of sale not subject to tax
ExemptionType	int	set to 0 - reserved for future use
InvoiceNumber	uint	matches user input in transaction data
Lines	int	For tax calculations based on line counts, it is the number of lines used in the calculation to produce the tax. For Federal taxes, the line count always matches the lines input. For all other taxes, it is 0.
Locations	int	number of locations in transaction data
LogCounter	int	Does not apply for bridge conference. Will always be 0.
Minutes	int	For tax calculations based on minutes, it is the number of minutes used in the calculation to produce the tax. For all other taxes, it is 0.
Optional	uint	matches user input in transaction data
Optional4	uint	matches user input in transaction data
Optional5	uint	matches user input in transaction data
Optional6	uint	matches user input in transaction data
Optional7	uint	matches user input in transaction data
Optional8	uint	matches user input in transaction data
Optional9	uint	matches user input in transaction data
Optional10	uint	matches user input in transaction data
OptionalAlpha1	string	matches user input in transaction data
PCode	uint	PCode for the reporting jurisdiction
RefundUncollect	double	amount of sale refunded or written off
SaleAmount	double	Taxable measure for transaction
ServiceLevelNumber	uint	matches user input in transaction data
SrvType	short	Service type used for transaction.
Surcharge	bool	true = tax considered a surcharge
TaxAmount	double	tax amount
TaxLevel	int	one of the TaxLevel enums
TaxRate	double	Tax rate
TaxType	short	one of the Avalara tax types
TransCharge	double	Charge for transaction
TransType	short	Transaction type used for transaction.

OptionalField		
Data Definition		
Property Name	**Type**	**Description**
OptionalKeyNo	short	Optional field key number (values from 1 to 10)
OptionalValue	string	Value for optional field (up to 150 characters)

OptionalKey

Data Definition

	Property Name	Type	Description
	OptionalKeyNo	short	Optional field key number (values from 1 to 10)
	OptionalKeyDesc	string	Description or label for optional field key.

ZipLookup

Data Definition

	Property Name	Type	Description
	Country	string	3 character ISO abbreviation
	County	string	County name
	City	string	City name
	State	string	State Abbreviation
	ZipCode	string	ZIP Code (5 digit)
	BestMatchFlag	Boolean	True = Best Match logic False = Exact Match
	LimitResults	Integer	Limit results for match to this value

ZipLookupResult

Data Definition

	Property Name	Type	Description
	InputMatchType	Best/Exact	Requested match type
	MatchCount	Integer	Number of matches returned
	MatchTypeApplied	Best/Exact	Match type used
	ResultsLimit	Integer	State Abbreviation
	LocationData	Object	Array of LocationData objects that match requested lookup

LocationData

Data Definition

	Property Name	Type	Description
	PCode	Long	PCode for location
	Country	string	3 character ISO abbreviation
	County	string	County name
	City	string	City name
	State	string	State Abbreviation

SafeHarborOverride

Data Definition

	Property Name	Type	Description
	OriginalFederalTam	double	TAM value specified as decimal between 0 and 1.
	NewFederalTam	double	TAM value specified as decimal between 0 and 1.
	SafeHarborType	short	TAM Override Type 1 = Cellular, 2 = VoIP, 4 = Paging

CommitData			
Data Definition			
Property Name	**Value**	**Type**	**Description**
DocumentCode	User-defined, 150 character limit, Alphanumeric	String	Document code for transactions to commit or uncommit.
Committed	True False	bool	Indicates if document code should be committed or uncommitted.
CustomerNumber	User-defined	String	Optional. 20 characters max.
InvoiceNumber	User-defined	Nullable<int>	Optional
Optional	User-defined	Nullable<int>	Optional
Optional4	User-defined	Nullable<int>	Optional
Optional5	User-defined	Nullable<int>	Optional
Optional6	User-defined	Nullable<int>	Optional
Optional7	User-defined	Nullable<int>	Optional
Optional8	User-defined	Nullable<int>	Optional
Optional9	User-defined	Nullable<int>	Optional
Optional10	User-defined	Nullable<int>	Optional
OptionalAlpha	User-defined	String	Optional. 20 characters max.
OptionalFields	User-defined	OptionalField[]	Array of OptionalField objects.

Note: The Bridge Conferencing feature is currently in development at this time. Please refrain from using this feature as well as the proposed solutions and APIs until further notice.

BridgeConferenceTransaction		
Data Definition		
Property Name	**Type**	**Description**
CustomerType	int	One of the CustomerType enums
BusinessClass	int	One of the BusinessClass enums
Sale	bool	true = Sale, false = Resale (Wholesale)
TransactionType	short	Not used. Can be 0/0.
ServiceType	short	
ServiceClass	int?	One of the ServiceClass enums
Date	DateTime	Date of transaction
Charge	double	Amount to tax **per participant**.
Incorporated	bool	true = address inside city limits false = unincorporated address
FederalExempt	bool	true = exempt federal level taxes
StateExempt	bool	true = exempt state level taxes
CountyExempt	bool	true = exempt county level taxes
LocalExempt	bool	true = exempt city/local taxes
FederalPCode	uint	federal PCode to exempt
StatePCode	uint	state PCode to exempt
CountyPCode	uint	county PCode to exempt
LocalPCode	uint	local PCode to exempt

	BridgeConferenceTransaction		
	Data Definition		
	Property Name	Type	Description
	Exclusions	Exclusion[]	Array of Exclusion objects to apply to this transaction
	Exemptions	TaxExemption[]	Array of TaxExemption objects to apply to this transaction
	CategoryExemptions	CategoryExemption[]	Array of CategoryExemption objects to apply this transaction
	ExemptionType	int	set to 0 - reserved for future use
	InvoiceNumber	uint	optional invoice or order number
	Optional	uint	optional field
	CustomerNumber	string	optional customer account number (up to 20 characters)
	CompanyIdentifier	string	optional company identifier (up to 20 characters)
	OptionalAlpha1	string	optional field up to 20 characters
	Optional4	uint	optional field
	Optional5	uint	optional field
	Optional6	uint	optional field
	Optional7	uint	optional field
	Optional8	uint	optional field
	Optional9	uint	optional field
	Optional10	uint	For live transactions, this field must indicate the billing cycle in YYYYMM format.
	AdjustmentMethod	int	for adjustments only (otherwise zero) One of the AdjustmentMethod enums
	BillingAddress	ZipAddress	ZipAddress object for Billing.
	BillingPCode	uint?	PCode for Billing.
	BillingNpaNxx	uint?	NpaNxx for Billing.
	BridgeAddress	ZipAddress	ZipAddress object for Bridge.
	BridgePCode	uint?	PCode for Bridge.
	BridgeNpaNxx	uint?	NpaNxx for Bridge.
	HostAddress	ZipAddress	ZipAddress object for Host. Optional.
	HostPCode	uint?	PCode for Host. Optional.
	HostNpaNxx	uint?	NpaNxx for Host. Optional.
	Participants	BridgeConferenceParticipant[]	Array of BridgeConferenceParticipant objects to apply to this transaction.
	Lines	int	For tax calculations based on line counts, it is the number of lines used in the calculation to produce the tax. For Federal taxes, the line count always matches the lines input. For all other taxes, it is 0.
	Locations	int	Number of customer locations
	Minutes	double	For tax calculations based on minutes, it is the number of minutes used in the calculation to produce the tax. For all other taxes, it is 0.
	bool	Debit	true = debit (prepaid)

BridgeConferenceTransaction

	Property Name	Type	Description
	DiscountType	int	One of the DiscountType enums
	FacilitiesBased	bool	true = facilities based
	Franchise	bool	true = franchise
	Lifeline	bool	true = lifeline customer
	Regulated	bool	true = regulated
	ServiceLevelNumber	uint	Optional field
	ProcessInvalidParticipant	bool	Flag indicating whether invalid participant jurisdiction should cause the transaction to fail (false) or whether the participant should be processed with greatest tax liability (true).
	ReturnParticipantTaxes	bool	Flag indicating whether detailed list of individual participant taxes should be returned along with the summarized taxes.

Data Definition row header above Property Name/Type/Description.

BridgeConferenceParticipant

Data Definition

	Property Name	Type	Description
	ParticipantAddress	ZipAddress	ZipAddress object for Participant – all participants must use ZipAddress values if any participant uses ZipAddress.
	ParticipantNpaNxx	uint?	NpaNxx for Participant – all Participants must use NpaNxx values if any participant uses NpaNxx.
	ParticipantPCode	uint?	PCode for Participant – all Participants must use PCode values if any participant uses PCode.
	ParticipantRef	string	Alpha-numeric reference that will tie participant in request to participant taxes in result, if ReturnParticipantTaxes is true. Max 127 bytes.

BridgeConferenceResults

Data Definition

	Property Name	Type	Description
	ParticipantResults	BridgeConferenceParticipantResult[]	Array of BridgeConferenceParticipantResult objects for transaction. Only returned if ReturnParticipantTaxes is true.
	Taxes	TaxData[]	An array of TaxData objects that contain the information about the summarized taxes applied for all participants.

BridgeConferenceParticipantResult

Data Definition

	Property Name	Type	Description
	ErrorCode	string	Error code for participant tax calculation (as applicable).

BridgeConferenceParticipantResult			
Data Definition			
	Property Name	Type	Description
	ParticipantRef	string	Participant reference from request.
	ParticipantTaxes	TaxLogDataV914[]	An array of TaxLogDataV914 objects that contain the information about the taxes for each individual participant.
	TransactionType	short	Transaction type used for participant.
	ServiceType	short	Service type used for participant.

14. Web Service Enumeration Definitions

14.1 Specifying Tax Level

TaxLevel			
Data Definition			
	Name	Value	Description
	Federal	0	Federal level tax
	State	1	State level tax
	County	2	County level tax
	Local	3	Local (city) tax
	Unincorporated	4	County unincorporated tax
	Other	5	Grouped tax result
	State_County_Local	6	Grouped tax result
	County_Local	7	Grouped tax result

14.2 Specifying Calculation Type

CalculationType			
Data Definition			
	Name	Value	Description
	Rate	1	tax based on charge amount
	Fixed	2	fixed tax
	PerMinute	3	tax based on minutes
	PerLine	4	tax based on number of access lines
	SelfTaxingRate	5	tax that taxes itself
	PerBracket	6	fixed tax with accumulated brackets
	FixedOnTier	7	fixed tax based on final bracket

14.3 Specifying Adjustment Method

In certain cases, when the rate of a tax will change based upon the charge or the number of lines in the transaction, clients may wish to make an adjustment at a different rate than would normally apply for the transaction.

For example, if the client purchased 20 lines at a rate of $0.67 per line and the credit or adjustment is for 5 lines at a rate of $1.25 per line, then usage of LEAST for the adjustment method would refund the 5 lines at the least favorable rate for the brackets making sure the client was not over compensated.

If unsure, use the value "Default".

AdjustmentMethod		
Data Definition		
Name	Value	Description
Default	0	tax brackets applied normally
LeastFavorableRate	1	tax brackets applied to produce smallest tax refund
MostFavorableRate	2	tax brackets applied to produce largest tax refund

14.4 Specifying Customer Type

This field is used to specify the type of customer involved in the transaction. The customer type is selected from one of the following four Customer Types.

CustomerType		
Data Definition		
Name	Value	Description
Residential	0	When transactions are made by a customer for home use.
Business	1	When transactions are made at a place of business.
SeniorCitizen	2	When transactions made by a customer meeting the jurisdiction requirements to be considered a senior citizen and qualify for senior citizen tax breaks.
Industrial	3	When transactions are made at an industrial business.

14.5 Specifying Business Class

The Business Class Indicator field is used to specify if the business making the transaction is an Incumbent Local Exchange Company (ILEC) or a Competitive Local Exchange Company (CLEC). Applies to Customer.

This setting can also be viewed as "you are ILEC" or 'you are not ILEC." If neither class applies, use the default value "CLEC".

The business class only impacts Telecom transactions in certain jurisdictions, such as Oregon.

BusinessClass		
Data Definition		
Name	Value	Description
ILEC	0	Incumbent Local Exchange Company
CLEC	1	Competitive Local Exchange Company

14.6 Specifying Service Class

The Service Class Indicator is provided to delineate the Primary activity of an organization as either Long Distance or Local Service.

The service class only impacts Telecom transactions in certain jurisdictions, such as New York.

ServiceClass		
Data Definition		
Name	Value	Description
PrimaryLocal	0	Primary Local Service providers are carriers vending their services with over 50% of the gross business activities in Local Service revenue.
PrimaryLongDistance	1	Primary Long Distance providers are carriers vending their services with over 50% of the gross business activities in Long Distance revenue.

14.7 Specifying Discount Type

AFC has an additional table that stores discount types by state with an "allow ability" indicator. The adjustment functions have arguments for the discount type which look up the discount type from the table to determine whether to apply taxes or not.

Discounts may or may not be taxed within each state. When a discount is taxed, the customer receives a tax benefit commensurate with the amount of the discount (i.e., if the customer gets $5 off on a transaction subject to a 5% tax, the customer pays $0.25 less in tax than they would have). When a discount is not taxed, the customer receives non tax benefit from the discount. Whether a discount is taxed or not depends on the type of discount and the rules in a particular tax jurisdiction.

Discount types only apply to adjustment transactions. Refer to **Section 4.7.1 Discount Adjustments in the AFC Telecom User Manual** for more information on calculating adjustments.

DiscountType		
Data Definition		
Name	Value	Description
None (Default)	0	Discount type not applicable. This should be the value used when the listed options do not apply. Note that choosing a value other than **None** can reduce the refund if the jurisdiction has determined that taxes are not refundable for the discount type specified.
RetailProduct	1	An amount subtracted from the original price to arrive at a lower price.
ManufacturerProduct	2	A discount of the total amount reimbursed to either the retailer or the customer by the manufacturer.
AccountLevel	3	A stand alone discount that is not applied against any service but instead as a stand alone product.

DiscountType		
Data Definition		
Name	Value	Description
Subsidized	4	A discount caused exclusively in telephone service where the telephone provider provides a service to a lifeline eligible customer. The discount will be on the local exchange service.
Goodwill	5	The total discount of a service that is recorded for accounting purposes but never billed to a customer.

14.8 Specifying Exemption Type

- Specifying the exemption type is for reporting purposes only and does not impact tax calculations.
- Spaces and capitalization do not matter in Exemption Type value.
- The **bolded** text is sufficient, additional characters are ignored.
- The optional numeric ID can be used. For example:
  Non Profit Sales = nonprofitsales = nonprofit = 6
  Fed Coupons WIC Vouchers = Fed Coupons = FEDCOUPONS = 4

ExemptionType		
Name	Value	Description
None (Default)	0	Exemption Type not applicable.
Fed Sales Supremacy	1	Sales to the Federal government exempt based on supremacy.
State Local Gov Sales	2	Sales to the State or Local government in the state that you operate in.
Sales for Resale	3	Sales that are designated for resale.
Fed Coupons WIC Vouchers	4	Sales at retail for which Federal government coupons or vouchers under the supplemental feeding for women, infants and children program are used as payment.
Reduced Food Rate	5	Tax imposed on food that is taxed at a reduced rate.
Non Profit Sales	6	Sales made by or to not-for-profit civic, social, service or fraternal organizations.
Public School Sales	7	Sales made by or to all elementary and secondary schools operated at public expense in their educational functions and activities.
Religious Charitable Sales	8	Sales made by or to religious and charitable organizations and institutions in their religious, charitable or educational functions and activities.
Prescription Drug Sales	9	Drugs which may be legally dispensed by a licensed pharmacist only upon a lawful prescription of a practitioner licensed to administer those items.
Prostho Ortho Devices	10	Prosthetic or orthopedic devices as defined on January 1, 1980, by the Federal Medicare program pursuant to Title XVIII of the Social Security Act of 1965.
Insulin Sales	11	All sales of insulin.
Interstate Foreign Sales	12	Sales in commerce between various states or between any foreign country (Export Sales).

ExemptionType		
Name	Value	Description
Ingredient Component Parts	13	Sale of ingredient or component parts.

14.9 Specifying Lines

When local service is provided, a transaction should be generated with the Lines field populated with the number of lines the customer subscribes to. AFC uses this information for generation of per line taxes usually associated with local E911 charges and local telecommunications relay service taxes and other assorted taxes.

Transaction Fields	
Column Name	Description
Lines	Numeric : integer >= 0

14.10 Specifying Sale Type

Sale should be used for retail transactions. Resale should be used for wholesale transactions. Telecommunication companies are taxed on transactions made by their clients, which in some cases can be passed on or "resold" to their customers in part or in total. Resale is used to indicate whether the product or service transaction is final or if it is to be resold.

To have exempt taxes available for reporting, exemption type 3 (Sales For Resale) should be used in combination with Resale.

Sale Type	
Value	Description
Sale (Default)	Standard taxes apply
Resale	Only Resale/Wholesale taxes apply

14.11 Specifying Incorporated

The Incorp field is used to specify whether the customer involved in this transaction is inside or outside of the Local level designated as their location. The tax may or may not be affected by this designator depending upon whether or not the local level has taxes which would apply to the transaction/service type pair. Marking transactions as Incorporated = False will usually result in no local taxes returned. If not sure, the value True is the recommended setting to be used.

Incorporated Flag	
Value	Description
True (Default)	Jurisdiction is inside an incorporated location.

| False | Jurisdiction is outside an incorporated location. |

15. Monthly Update

The AFC SaaS Pro monthly update is available at approximately 4:00 PM Central time on the day before the last business day of each month. It contains updated tax information and database files and is available at the Avalara Support web site. The update contains changes resulting from ongoing research and development, providing the most current and efficient tax-rating engine available.

Note: Historical and current month tax rates are not impacted by the updates. Updates are reflected in the subsequent month's tax rates.