



User Guide and API Reference

Overview

Version 1.8.0

3/19/2012

Table of Contents

User Guide and API Reference	1
Introduction	4
How the API Works.....	5
Supported Tools	6
The Services Available	7
Batch:	7
Single Record:.....	7
The Environments.....	8
Customer Test Environment (CTE):	8
Production Environment (WS):	8
Security	9
Getting Started	9
New Account Establishment:	9
AccountService (Web Service):.....	10
Account (Provider, Client, End User) Examples.....	11
Diagram – Provider w/pass through accounting:.....	11
Diagram – Provider w/single client accounting:.....	12
Diagram Single client accounting w/User Group Examples:.....	13
Products	14
Batch Processes:.....	14
Single Record Processes:.....	14
BATCH Processing Endpoints	15
AuthenticationService.....	15
BATCH Processing Endpoints (continued).....	16
OrderService.....	16
BATCH Processing Endpoints (continued).....	17
BATCH Processing Endpoints (continued).....	18
AccountService	18
BATCH Processing Endpoints (continued).....	19
AdminService.....	19

Web Service User Guide and API Reference

PaymentService (<i>Has been replaced by the PaymentProfileService</i>)	19
PaymentProfileService	19
SINGLE Record Processing Endpoints	20
AuthenticationService	20
CASSService	21
Payment Services	22
1. Credit Card Payment (End User):	22
2. Card Payment (TEC Mailing Customer ‘provider’):	22
3. Invoice (TEC Mailing Customer ‘provider’):	22
4. Invoice (Provider’s Client ‘customer’):	22
NCOA ^{Link} Processing	23
HTTP POST example for ‘End User’ to access ExpressPAF (Test Environment):	23
CASS Output (Batch)	24
NCOA ^{Link} Output (Batch)	24
Presort Output (Batch)	24
Output Report Definitions	25
Address Hygiene	25
Presort	25
Simple Use of Methods (BATCH Process)	27
CreateOrderRequest (OrderService - BATCH Process)	28
FieldMapping (OrderService - BATCH Process)	29
UploadFile (OrderService - BATCH Process)	30
ExecuteOrder (OrderService - BATCH Process)	31
GetOrder (OrderService - BATCH Process)	32
GetOrderOutput (OrderService - BATCH Process)	33
CreateAccount (AccountService - BATCH Process)	34
CreateUserGroup (AccountService - BATCH Process)	36

Introduction

The TEC Mailing Solutions Web Service API is an Application Programming Interface, which allows you to connect your application with TEC Mailing Solutions' web interfaces. Using the Web Service API, you can seamlessly process address files through a number of services directly in your application.

The TEC Mailing Solutions Web Service API is a SOAP-based web service. Some of the advantages of offering integration using a web service include:

- Platform independence — Any application that can send and receive SOAP messages can communicate with the Web Service API. Because the Web Service API is built using open standards, you can choose any technology that suits your needs (e.g. J2EE, .NET, PHP, ASP, etc.) for integrating with the TEC Mailing Solutions Web Service API.
- Ease of integration — Communicating with a web service is simple. Your application builds a SOAP request message encoding your transaction, sends it via HTTPS to the web service, and waits for a SOAP response message, which contains your transaction's status. Since SOAP and HTTPS are designed to be lightweight protocols, building requests and parsing responses is a straightforward. Furthermore, rarely do you have to do this manually, since there are a number of libraries available in almost every technology. In general, building a SOAP request and handling the response is reduced to a few lines of code.
- Security — All communication between your application and TEC Mailing Solutions Web Service API is SSL-encrypted. Your application has a client certificate, which identifies it uniquely with the web service. The Web Service API holds a server certificate, which your application checks to ensure that it is communicating with the Web Service API. The Web Service also requires HTTP basic authorization (user name and password) in order to communicate with the web service. These security mechanisms guarantee that the transaction data sent to TEC Mailing Solutions Web Service API stays private and is available only to your application.

This document will assist you in integrating your application with the Web Service API, and provide you with a brief summary of the Web Service API solution feature set.

How the API Works

The following section describes the process of creating a job and submitting data through the Web Service API.

In most cases, a customer starts the overall communication process by identifying a particular process to be performed on their address data (example NCOA^{Link} and Dedupe) in your application or online environment. Your application needs to send a transaction to TEC Mailing Solutions to establish a job. Having received the transaction, the TEC Mailing Solutions Web Service API forwards it to the application servers within TEC Mailing Solutions' data center. Based on the defined processes, your application receives a status update or an error response from the Web Service API.



Web service interfaces are designed using the Web Service Definition Language (WSDL). The WSDL file for the Web Service API is located here:

See: [End Points](#)

Note: Your user ID and password are not required to view the WSDL but they are required to access the TEC Mailing Solutions Web Service API.

The WSDL file defines the operations offered by the Web Service API, their request and response parameters, and how to call the operations. The TEC Mailing Solutions Web Service API WSDL file defines many operations, called by sending a SOAP request to the following URL:

Web Service User Guide and API Reference

Customer Test Environment:

See: End Points for wsdl

This operation takes an XML-encoded transaction as a request and returns an XML-encoded response. Depending on the tools you use to integrate with the Web Service API, you may need to provide the URL for the WSDL file. If so, you must tell your tool that the communication is SSL-enabled, provide your client certificate, and accept the server certificate as trusted. The process for this depends upon your tool. Consult the documentation for your tool for details.

Supported Tools

The TEC Mailing Solutions Web Service API uses HTTPS and SOAP to communicate with your applications. As such, it is completely platform independent. The choice of languages, frameworks, or tools to integrate with the Web Service API is up to you.

TEC Mailing Solutions has tested the Web Service API with the following tools:

- ASP.NET Framework
- C#.NET Framework
- VB.NET Framework
- PHP

While you can use any tools to integrate with the API, these are the tools TEC Mailing Solutions has tested. Integrating with the TEC Mailing Solutions Web Service API using other tools is outside the scope of this document.

The Services Available

TEC Mailing Solutions has two high level methodologies available: Batch and Single Record. Within each of these methodologies are a variety of services to clean, update, and sort address lists.

Batch:

The Batch method is intended to accommodate a complete file of address records. In the case of address hygiene there is a minimum of 100 records and a maximum file size of 50MB. For the presort process there is a minimum based on the class of mail (1st Class = 500 and STD = 200) and a maximum file size of 50MB. The services available in the batch method are:

1. CASS Alone
2. CASS + NCOALink
3. CASS + NCOALink + Duplicate Detection
4. Presort Alone (Requires CASS certified data as input)
5. Presort + CASS
6. Presort + CASS + NCOALink
7. Presort + CASS + NCOALink + Duplicate Detection

Single Record:

The Single Record method is intended to accommodate one record at a time. The services available in the Single Record method are:

1. CASS Alone

Web Service User Guide and API Reference

The Environments

TEC Mailing Solutions has two environments that are available for users 'providers' to utilize its Web Service API: Customer Test Environment and Production Environment. There are different endpoints and wsdl's for each location.

Customer Test Environment (CTE): The CTE is established in a location to allow new customers to build and test their applications utilizing the TEC Mailing Solutions Web Service API. This environment is not intended to process any live production. In this environment, only subsets of the actual output will be returned. NOTE: All jobs processed within the CTE environment will NOT be invoiced or billed to a credit card.

Production Environment (WS): The production environment is the actual processing environment supporting the full host of services and methods of the TEC Mailing Solutions Web Service API. NOTE: All jobs processed within the production environment will be invoiced or billed to a credit card.

Security

The customer test environment is intended to provide developers a location to establish connections, test WS calls and processing code. It is not an environment to process live files. Therefore, this environment has limited security and developers should treat as a “test” environment. To access the customer test environment the developer will only be required to have a user account, password, and UserDatabaseID. All of these will be provided by TEC Mailing Solutions.

The production environment will require the provider (user\developer of the actual web service) a provider UserDatabaseID, Channel ID, and Product ID. Each user of the end product will be required to have a user account and password that is communicated through the web service.

Getting Started

How do I get started using TEC Mailing Solutions Web Service API? All users of the Web Service API must contact TEC Mailing Solutions to obtain a customer account. This process may involve a credit approval as well as the signing of a “Sales Contractor Agreement”.

New Account Establishment:

Currently, all new users ‘providers’ whom are to use and develop under the TEC Mailing Solutions Web Service API must be manually created by TEC Mailing Solutions. End users who are created under ‘Customers’ can be created by directing the user\developer via the AccountService.

Upon creation of a new account a developer (provider or customer) will be given the following values to describe the product being developed. These values are required for the purpose of end user account creation and must be implemented to access the Web Service API:

1. userdatabaseid = A GUID identifying the provider or customer’s account
2. channelid = A GUID identifying the service and/or business channel that is being utilized.
3. productid = A GUID identifying the actual product/service group that is being utilized.

Additionally in order to test the web services an end user account must be established by TEC Mailing Solutions. This account will be comprised of the following credentials:

1. User Login
2. User Password

End user account creation is available through TEC Mailing Solutions’ AccountService web service. The AccountService supports the creation, and edit of end user accounts within a given userdatabaseid.

Web Service User Guide and API Reference

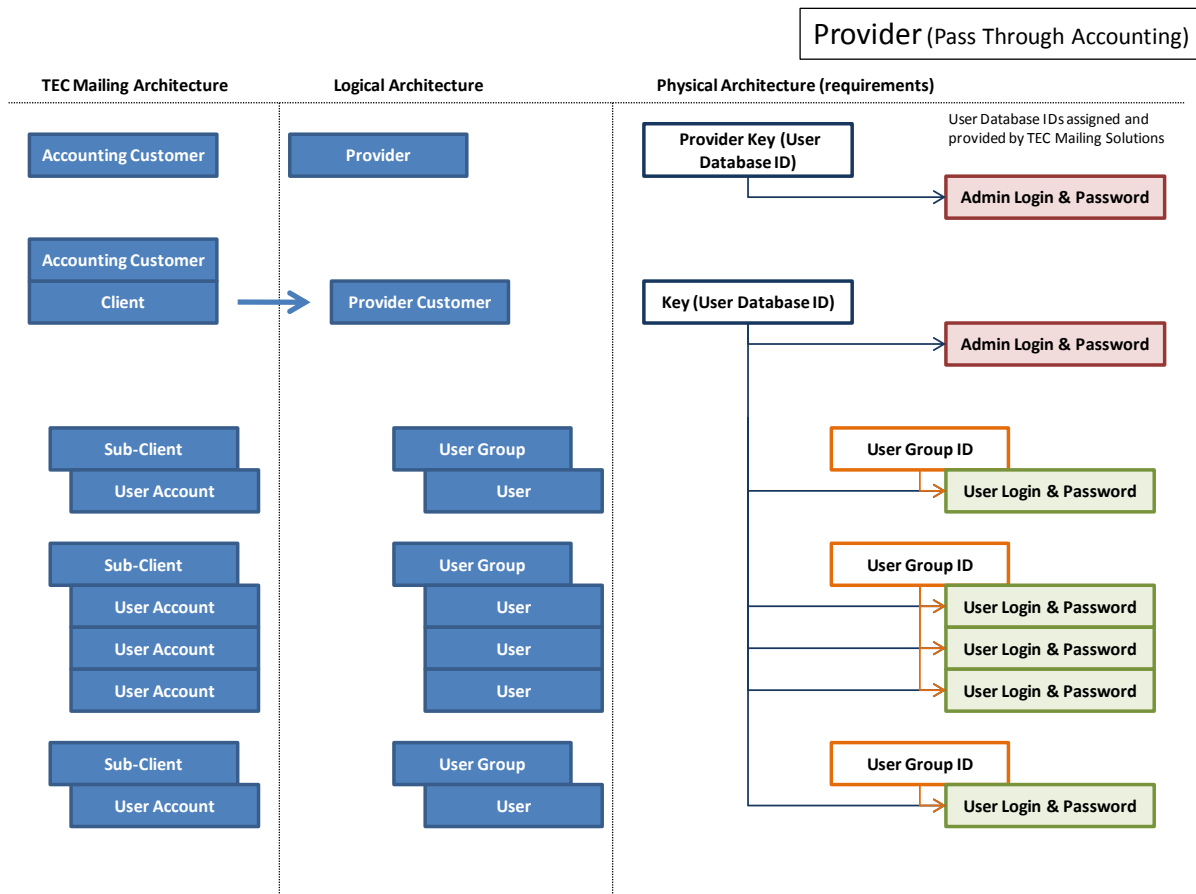
AccountService (Web Service):

The AccountService provides a number of methods to create, edit, and delete “User Groups” and/or “User Accounts”. There are a number of ways that accounts can be constructed and managed. The more control that the provider\developer can have on the structure of the account information the better. As part of the end user account process. TEC Mailing Solutions places the responsibility on the provider\developer to display, reference or require the viewing of TEC Mailing Solutions’ end users’ terms and conditions information provided at: [Http://www.tecmailapps.com/terms.aspx](http://www.tecmailapps.com/terms.aspx). A number of typical user an user group scenarios are provided below in two graphics.

Account (Provider, Client, End User) Examples

The following three diagrams detail the options around the provider (developer), the Client (Customer whom may be paying for the services, or managing the pricing), User Groups (any group or end users associated with each other. Example: Company), and the End Users.

Diagram - Provider w/pass through accounting: (Provider\Developer that has each of its clients paying for services through TEC Mailing Solutions).



Web Service User Guide and API Reference

Diagram – Provider w/single client accounting: (Provider\Developer that is directly paying for services through TEC Mailing Solutions).

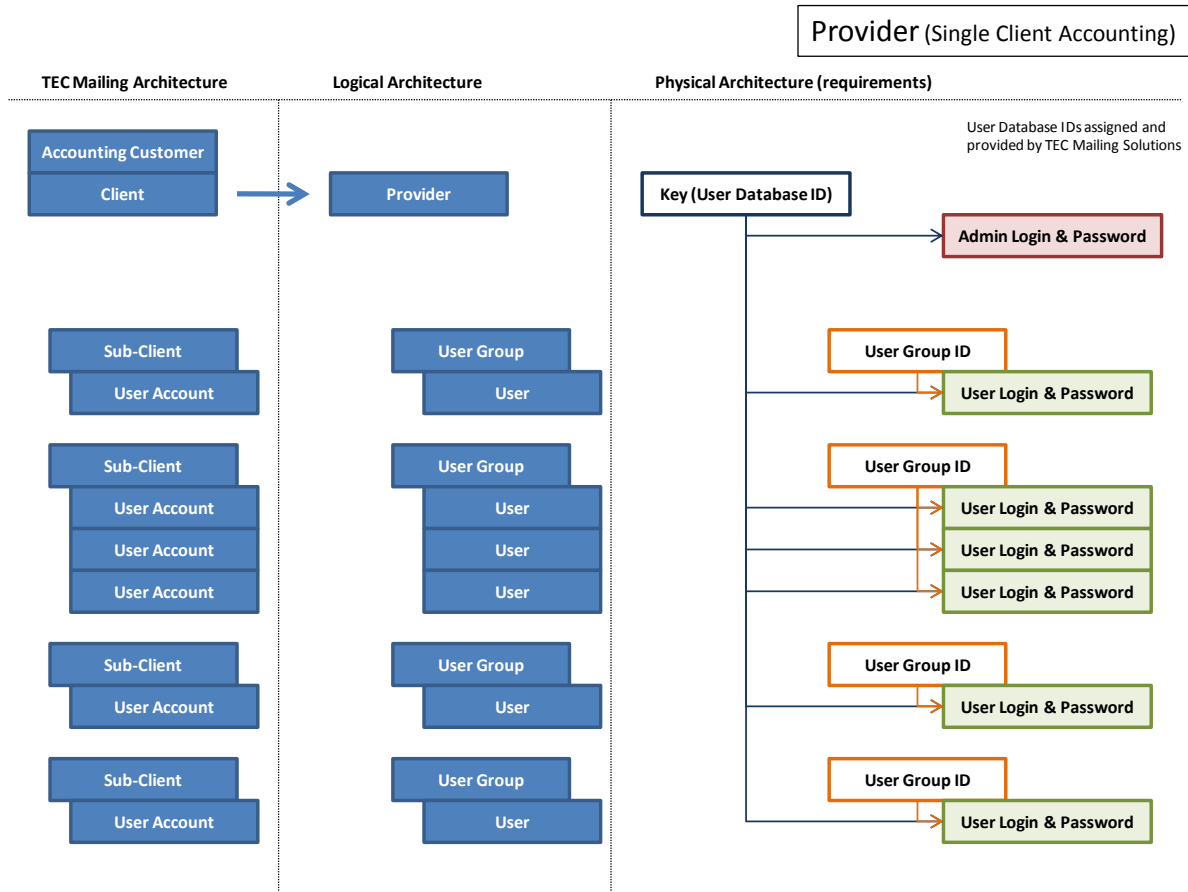
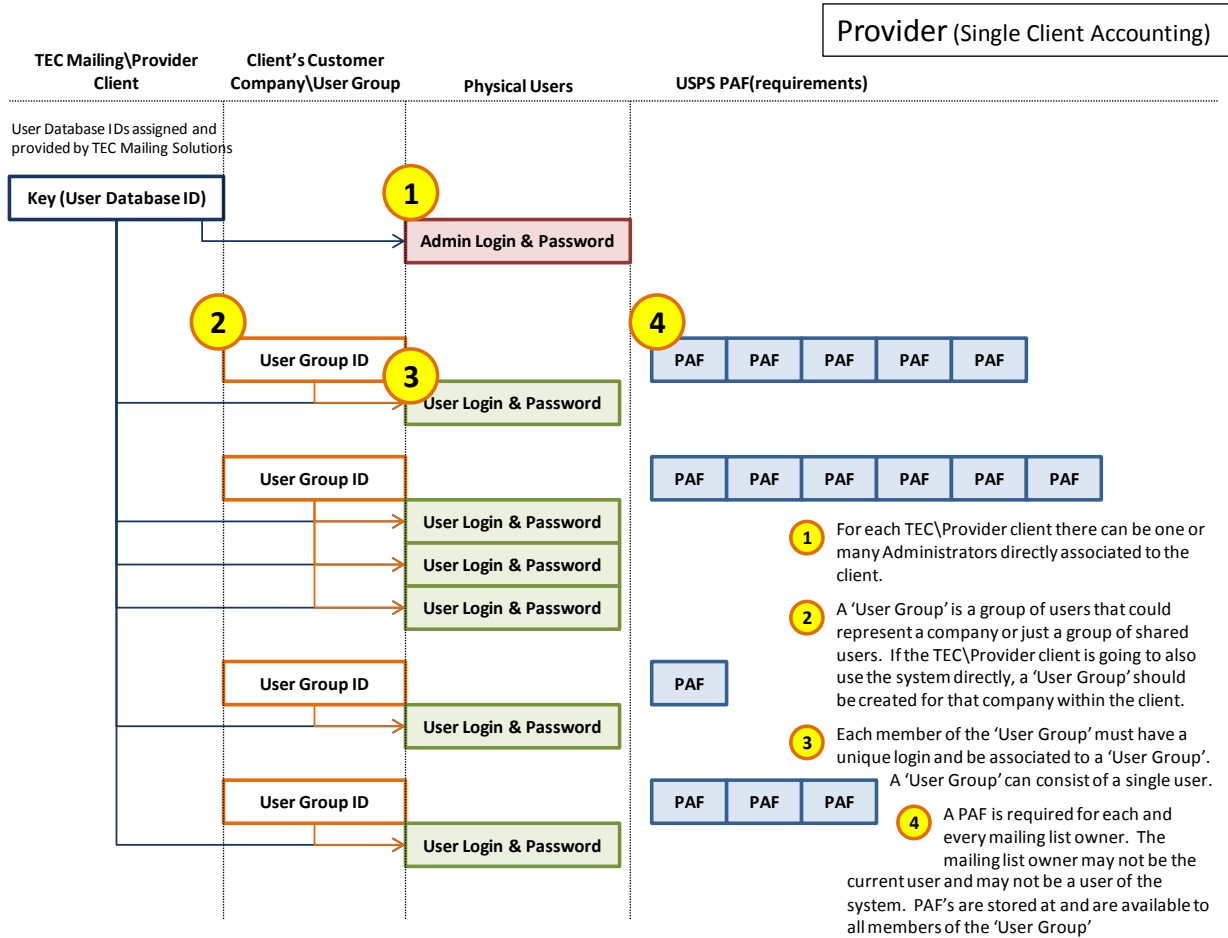


Diagram Single client accounting w/User Group Examples: (Provider\Developer that is directly paying for services through TEC Mailing Solutions).



Products

TEC Mailing Solutions provides currently two products: Batch and Single Record. The end points are lists in the next sections and more information can be found for these services at:

Batch Processes:

Production (Version 1)

http://www.tecmailing.com/TEC_batch_process_services_PRO_V1.html

Production (Version 2)

http://www.tecmailing.com/TEC_batch_process_services_PRO_V2.html

Customer Test Environment (CTE) (Version 1)

[http://www.tecmailing.com/TEC_batch_process_services\(CTE\).html](http://www.tecmailing.com/TEC_batch_process_services(CTE).html)

Customer Test Environment (CTE) (Version 2)

[http://www.tecmailing.com/TEC_batch_process_services\(CTE2\).html](http://www.tecmailing.com/TEC_batch_process_services(CTE2).html)

Single Record Processes:

http://www.tecmailing.com/TEC_SINGLE_CASS_PROCESS_SERVICES.html

BATCH Processing Endpoints

These endpoints may not currently be active and may change before a production release.

AuthenticationService

Customer Test Environment (Version 1):

<https://ws.cte.tecapps.com/AuthenticationService.svc>

<https://ws.cte.tecapps.com/AuthenticationService.svc?wsdl>

Customer Test Environment (Version 2):

<https://secure.cte.tecapps.com/AuthenticationService.svc>

<https://secure.cte.tecapps.com/AuthenticationService.svc?wsdl>

Production Environment (Version 1):

<https://tecwsapps.com/AuthenticationService.svc>

<https://tecwsapps.com/AuthenticationService.svc?wsdl>

Production Environment (Version 2):

<https://streaming.tecwsapps.com/AuthenticationService.svc>

<https://streaming.tecwsapps.com/AuthenticationService.svc?wsdl>

BATCH Processing Endpoints (continued)

These endpoints may not currently be active and may change before a production release.

OrderService

Customer Test Environment (Version 1):

<https://ws.cte.tecapps.com/OrderService.svc>

<https://ws.cte.tecapps.com/OrderService.svc?wsdl>

Customer Test Environment (Version 2):

<https://secure.cte.tecapps.com/OrderService.svc>

<https://secure.cte.tecapps.com/OrderService.svc?wsdl>

Production Environment (Version 1):

<https://tecwsapps.com/OrderService.svc>

<https://tecwsapps.com/OrderService.svc?wsdl>

Production Environment (Version 2):

<https://streaming.tecwsapps.com/OrderService.svc>

<https://streaming.tecwsapps.com/OrderService.svc?wsdl>

BATCH Processing Endpoints (continued)

These endpoints may not currently be active and may change before a production release.

PAFService

Customer Test Environment (Version 1):

<https://ws.cte.tecapps.com/PAFService.svc>

<https://ws.cte.tecapps.com/PAFService.svc?wsdl>

Customer Test Environment (Version 2):

<https://secure.cte.tecapps.com/PAFService.svc>

<https://secure.cte.tecapps.com/PAFService.svc?wsdl>

Production Environment (Version 1):

<https://tecwsapps.com/PAFService.svc>

<https://tecwsapps.com/PAFService.svc?wsdl>

Production Environment (Version 2):

<https://streaming.tecwsapps.com/PAFService.svc>

<https://streaming.tecwsapps.com/PAFService.svc?wsdl>

BATCH Processing Endpoints (continued)

These endpoints may not currently be active and may change before a production release.

AccountService

Customer Test Environment (Version 1):

<https://ws.cte.tecapps.com/AccountService.svc>

<https://ws.cte.tecapps.com/AccountService.svc?wsdl>

Customer Test Environment (Version 2):

<https://secure.cte.tecapps.com/AccountService.svc>

<https://secure.cte.tecapps.com/AccountService.svc?wsdl>

Production Environment (Version 1):

<https://tecwsapps.com/AccountService.svc>

<https://tecwsapps.com/AccountService.svc?wsdl>

Production Environment (Version 2):

<https://streaming.tecwsapps.com/AccountService.svc>

<https://streaming.tecwsapps.com/AccountService.svc?wsdl>

BATCH Processing Endpoints (continued)

These endpoints may not currently be active and may change before a production release.

AdminService

Customer Test Environment (Version 1):

<https://ws.cte.tecapps.com/AdminService.svc>

<https://ws.cte.tecapps.com/AdminService.svc?wsdl>

Customer Test Environment (Version 2):

<https://secure.cte.tecapps.com/AdminService.svc>

<https://secure.cte.tecapps.com/AdminService.svc?wsdl>

Production Environment (Version 1):

<https://tecwsapps.com/AdminService.svc>

<https://tecwsapps.com/AdminService.svc?wsdl>

Production Environment (Version 2):

<https://streaming.tecwsapps.com/AdminService.svc>

<https://streaming.tecwsapps.com/AdminService.svc?wsdl>

PaymentService (*Has been replaced by the PaymentProfileService*)

PaymentProfileService

Customer Test Environment (Version 2):

<https://tecsecurepay.cte.tecapps.com/PaymentProfileService.svc>

<https://tecsecurepay.cte.tecapps.com/PaymentProfileService.svc?wsdl>

Production Environment (Version 2):

<https://www.tecsecurepay.com/PaymentProfileService.svc>

<https://www.tecsecurepay.com/PaymentProfileService.svc?wsdl>

SINGLE Record Processing Endpoints

These endpoints may not currently be active

AuthenticationService

Customer Test Environment (Version 1):

<https://ws.cte.tecapps.com/AuthenticationService.svc>

<https://ws.cte.tecapps.com/AuthenticationService.svc?wsdl>

Customer Test Environment (Version 2):

<https://secure.cte.tecapps.com/AuthenticationService.svc>

<https://secure.cte.tecapps.com/AuthenticationService.svc?wsdl>

Production Environment (Version 1):

<https://tecwsapps.com/AuthenticationService.svc>

<https://tecwsapps.com/AuthenticationService.svc?wsdl>

Production Environment (Version 2):

<https://streaming.tecwsapps.com/AuthenticationService.svc>

<https://streaming.tecwsapps.com/AuthenticationService.svc?wsdl>

Web Service User Guide and API Reference

CASSService

Customer Test Environment (Version 1):

<https://ws.cte.tecapps.com/CASSService.svc>

<https://ws.cte.tecapps.com/CASSService.svc?wsdl>

Customer Test Environment (Version 2):

<https://secure.cte.tecapps.com/CASSService.svc>

<https://secure.cte.tecapps.com/CASSService.svc?wsdl>

Production Environment (Version 1):

<https://tecwsapps.com/CASSService.svc>

<https://tecwsapps.com/CASSService.svc?wsdl>

Production Environment (Version 2):

<https://streaming.tecwsapps.com/CASSService.svc>

<https://streaming.tecwsapps.com/CASSService.svc?wsdl>

Payment Services

Depending on the environment, TEC Mailing Solutions supports four methods of payment for services. The available payment method for each end user can be retrieved by calling the *OrderService.GetPaymentMethods*.

1. **Credit Card Payment (End User):** TEC Mailing Solutions supports the passing of end user credit card data for the payment process on a job by job basis. Though this is the easiest method, it should be understood that the information on the end credit card owners bill will reflect "TEC Mailing Solutions"

Under this method the *payment.PaymentMethod* should be set to *CreditCard*.

2. **Card Payment (TEC Mailing Customer 'provider'):** TEC Mailing Solutions supports the passing of provider credit card data for the payment process on a job by job basis.

Under this method the *payment.PaymentMethod* should be set to *CreditCard*.

3. **Invoice (TEC Mailing Customer 'provider'):** TEC Mailing Solutions supports the ability to invoice weekly the provider for services (jobs) performed within the prior 7 days. Under this method the *payment.PaymentMethod* should be set to *Terms*.

4. **Invoice (Provider's Client 'customer'):** (actually these customers become TEC Mailing Solutions customer). TEC Mailing Solutions supports the ability to invoice weekly the provider for services (jobs) performed within the prior 7 days. Under this method the *payment.PaymentMethod* should be set to *Terms*.

NCOALink Processing

TEC Mailing Solutions is a licensed provider of the USPS® NCOALink product. NCOALink provides the user with address move information for records that match the USPS NCOALink directories. There are a number of requirements that are not found with other services provided by TEC Mailing Solutions.

1. Minimum order requirement: There must be a minimum of 100 unique name and address combinations within any file submitted for the NCOA service.
2. Prior to submitting data through the NCOA service, the end user MUST obtain a USPS Process Acknowledgement Form (PAF). This PAF must be signed by the actual end list owner and the broker of the service. TEC Mailing Solutions provides an electronic means to perform all of these requirements at www.ExpressPAF.com. Once a PAF has been filled out and electronically signed by the end user and the broker, the PAF will be able to be referenced in the PAFService web service.
3. Reference a PAFDocumentID from the PAFService. This PAFDocumentID must be included in the AddressHygiene Datatype.

For more information on the USPS Process Acknowledgement Form and the user experience at ExpressPAF, visit [http://www.tecmailing.com/PAF_USER_EXPERIENCE\(20110601\).pdf](http://www.tecmailing.com/PAF_USER_EXPERIENCE(20110601).pdf)

HTTP POST example for 'End User' to access ExpressPAF (Test Environment):

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<body onLoad="javascript: document.forms[0].submit();">
<form action="http://www.tecmailsol.com:7878/SSO/Incoming.aspx" method="POST">
<input type="hidden" name="destinationpage" value="../summary.aspx">
<input type="hidden" name="useraction" value="transfer">
<input type="hidden" name="username" value="test1@xyzcompany.com">
<input type="hidden" name="password" value="test1!">
<input type="hidden" name="subclientname" value="XYZ Company CUSTOMER #1">
<input type="hidden" name="userdatabaseid" value="ffa9b27d-cxxc-4xxa-b481-2xx2756c825c">
<input type="hidden" name="channelid" value="03fccd9b-6d40-4xx8-bxxe-20exxbb778ea">
<input type="hidden" name="productid" value="bxxe6d12-xxbc-xx08-bcb1-axxc528a94f3">
</form>
</body>
</html>
```

Replace the `<form action="http://www.tecmailsol.com:7878/SSO/Incoming.aspx" method="POST">`
line with `<form action="https://www.expresspaf.com/SSO/Incoming.aspx" method="POST">` **for the production environment.**

CASS Output (Batch)

The output files from the CASS process based on an input file of "Sample.csv" are:

Sample_DefaultCASS.csv – Processing output csv

Sample_DefaultCASS.dmt – Processing output definition file

Sample_Input_File_Report.pdf – Quantity of input data report

Sample_CASS_Report_PS_FORM_3553.pdf - PS form 3553

Sample_CassReport.pdf – CASS report (Defines CASS coding Results)

File_layout_and_codes.xls – list of all fields and codes for all outputs

NCOALink Output (Batch)

The output files from the NCOA process based on an input file of "Sample.csv" are:

Sample_DefaultNCOA.csv – Processing output csv

Sample_DefaultNCOA.dmt – Processing output definition file

Sample_Input_File_Report.pdf – DIQ report

Sample_CASS_Report_PS_FORM_3553.pdf - PS form 3553

Sample_CassReport.pdf – CASS report (Defines CASS coding Results)

Sample_NCOALinkReport.pdf – NCOA report (Defines the returns and results for the NCOA process)

File_layout_and_codes.xls – list of all fields and codes for all outputs

TEC_SVC_PROV_RTD.txt – Required Service Provider documentation

Presort Output (Batch)

The output files from the Presort process based on an input file of "Sample.csv" are:

Sample_DefaultNCOA.csv – Processing output csv

Sample_DefaultNCOA.dmt – Processing output definition file

See details below for more presort output information

Output Report Definitions

Included in the output are a handful of standardized reports. Which reports are produced from each service is defined within the particular processes “output”. These are the definitions to these reports:

Address Hygiene

Sample_Input_File_Report.pdf: The “Input File” report is a simple report detailing the output quantity from one of this initial conversion steps. Its purpose is to report the number of records that have successfully imported.

Sample_CASS_Report_PS_FORM_3553.pdf: The USPS Form 3553 is required for each mailing claimed at all automation rates and all carrier route rates. The data recorded on Form 3553 refers to the date which the list has been processed. A mailing’s postage statement must be annotated in the block(s) provided to reflect the date when address matching and coding were performed.

Sample_CassReport.pdf: The “CASS Report” defines in detail how the list performed against the current USPS data. Critical information that is provided in this report includes: Qty coded to ZIP, ZIP+4, Carrier Route, and Delivery Point. Additionally details about type of delivery and Delivery Point Validation (DPV) are available. This report put simply defines how accurate your list is...

Sample_NCOALinkReport.pdf: The “NCOA Report” defines the results from matching the input list against the National Change of Address (NCOA) database. Critical information that is provided in this report includes: The number of matches to the NCOA database, the number of records returned with moves, the number of moves that could not be returned with new addresses, etc...

File_layout_and_codes.xls: A list of all fields and codes for all outputs

TEC_SVC_PROV_RTD.txt: Required Service Provider documentation for the NCOA^{Link} process

Presort

00000000_Mail_Sort_Listing.pdf: The “Mail Sort Listing” report defines how the mail should be prepared into bundles and/or containers.

00000000_USPS_Qualification_Report.pdf: The “USPS Qualification” report is very similar to the “Mail Sort Listing”. However, the qualification report is a required report for the USPS and defines more rate details and may to reflect exactly how the mail is prepared.

USPS_Postal_Statement_XXXXX.pdf: The “USPS Postal Statement” can end in either 3602 or 3600 depending on the class of mail being produced. This is the report that is required for the USPS and defines the amount of postage to be paid.

00000000_Tray_Tags.pdf: The “Tray Tags”

Web Service User Guide and API Reference

00000000_Sack_Tags.pdf: The “Sack Tags”

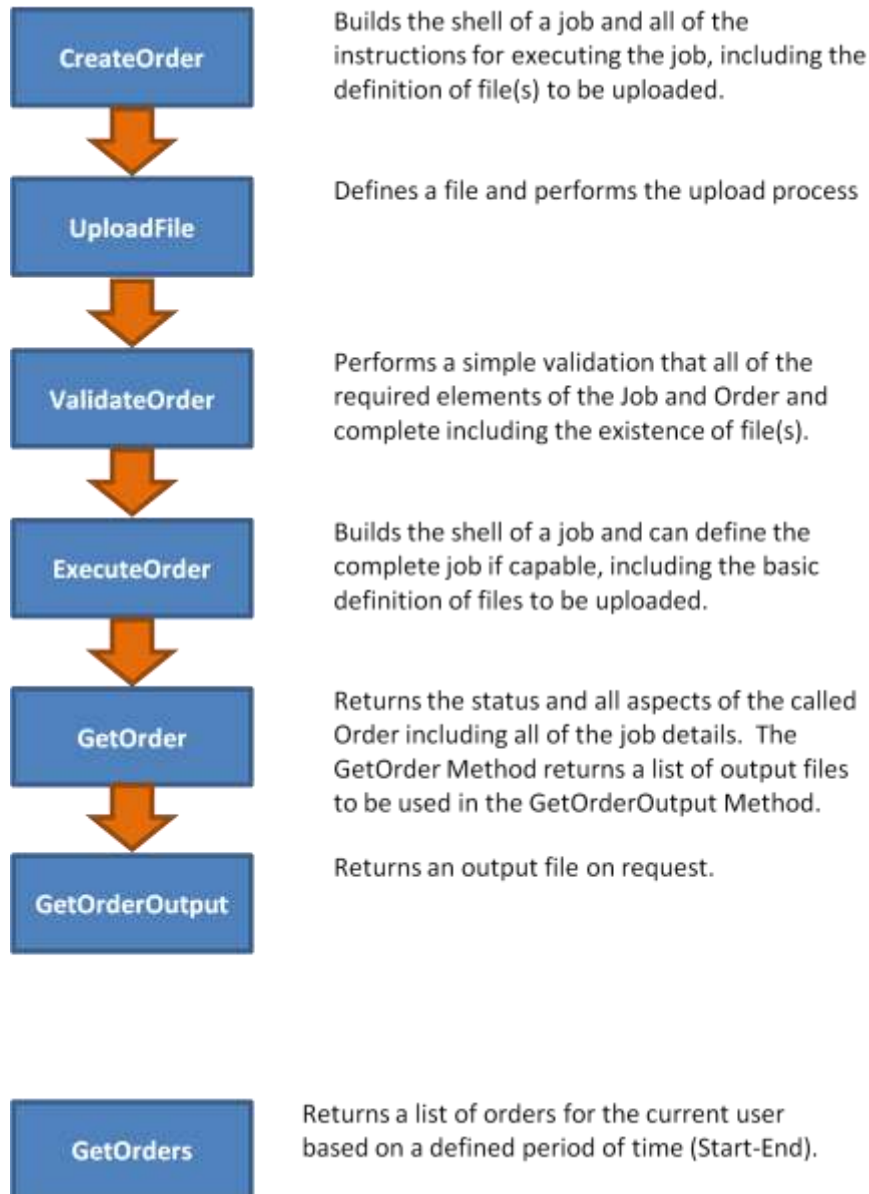
00000000_Pallet_Placards.pdf: The “Pallet Placards”

00000000_MotherToContainerRelationship.pdf: The “Mother to Container Relationship” report defines which trays are associated with which pallets if there is enough mail to create pallets.

MD_00000000.zip: The “MD” zipped file is the complete set of IDEAlliance Mail.dat files zipped as a compressed file.

Simple Use of Methods (BATCH Process)

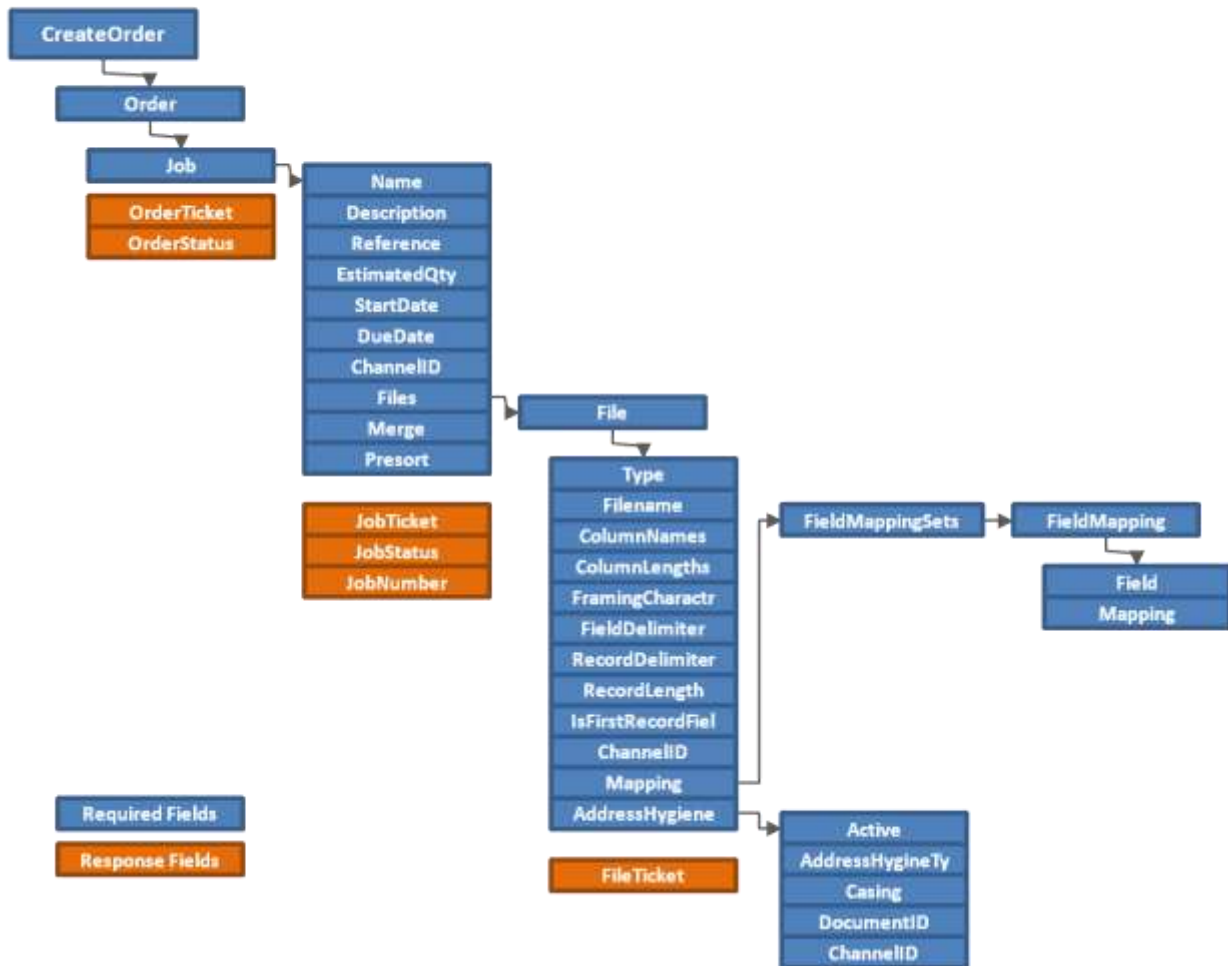
For a simplified purpose of demonstrating the use of the current methods, the diagram below visually describes the workflow process for the OrderService.



CreateOrderRequest (OrderService - BATCH Process)

For a simplified purpose of demonstrating the use of the CreateOrderRequest, the diagram below visually describes the requested fields and response values of the CreateOrder: *(This Method is the most comprehensive method of the OrderService. A complete order must be defined prior to uploading a file and executing any processing. The CreateOrder must contain all of the details about the file to be uploaded.)*

CreateOrderRequest - Simple Address Hygiene (No Merge)



FieldMapping (OrderService - BATCH Process)

For a simplified purpose of demonstrating the use of the FieldMapping:

NOTE: Field names in the list of Column Names "ColumnNames" must not contain any spaces.

FieldMapping



Field = one of TEC Mailing's predefined field names:

NAME – (Either Company or Name is Required): Should describe the full name of the addressee

COMPANY – (Either Company or Name is Required): The Company\Firm\Organization name

Address1 – (Required): Should be considered as the Primary Address Information

Address2 – (Not Required): Should be considered as the Secondary Address Information

Address3 - (Not Required): Should be considered as ancillary address information

City – (Required): The city name for the address

State – (Required): The state name for the address

Zip - (Required, however under certain circumstances where the zip is unknown, you can map this field to a blank field in the data to have the process post back a zip

Mapping = an input field name with the "db." prefix:

The field mapping can be a single field name with the "db." prefix or a combination of fields to create a single mapping. Example "db.first+db.last" to create the name field.

C# Example of FieldMappingSets, FieldMappingSet, FieldMappings, FieldMapping, Field, and Mapping

```
Mapping mapping = new Mapping();

FieldMappingSet[] fieldMappingSets = new FieldMappingSet[1];
FieldMappingSet fieldMappingSet = new FieldMappingSet();
fieldMappingSets[0] = fieldMappingSet;

FieldMapping[] fieldMappings = new FieldMapping[6];

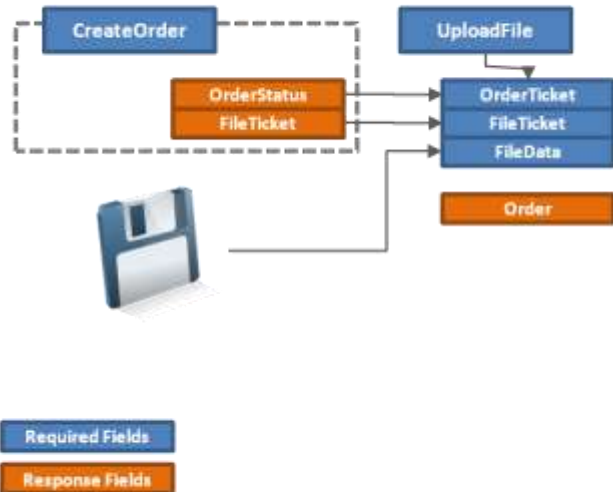
FieldMapping fieldMapping;

fieldMapping = new FieldMapping();
fieldMapping.Field = "NAME";
fieldMapping.Mapping = "DB.First_Name+DB.Last_Name";
fieldMappings[0] = fieldMapping;
```

UploadFile (OrderService - BATCH Process)

For a simplified purpose of demonstrating the use of the UploadFileRequest, the diagram below visually describes the requested fields and response values of the UploadFile Method:

UploadFileRequest



C# Example of the use of FileData and assignment of OrderTicket, FileTicket

```
UploadFileRequest uploadFileRequest = new UploadFileRequest();
uploadFileRequest.OrderTicket = orderticket;
uploadFileRequest.FileTicket = fileTicket;

try
{
    uploadFileRequest.FileData = System.IO.File.ReadAllBytes(_Inputfilename);
}
catch (Exception ex)
{
    MessageBox.Show("Error: Could not read file from disk. Original error: " + ex.Message);
}
```

Web Service User Guide and API Reference

ExecuteOrder (OrderService - BATCH Process)

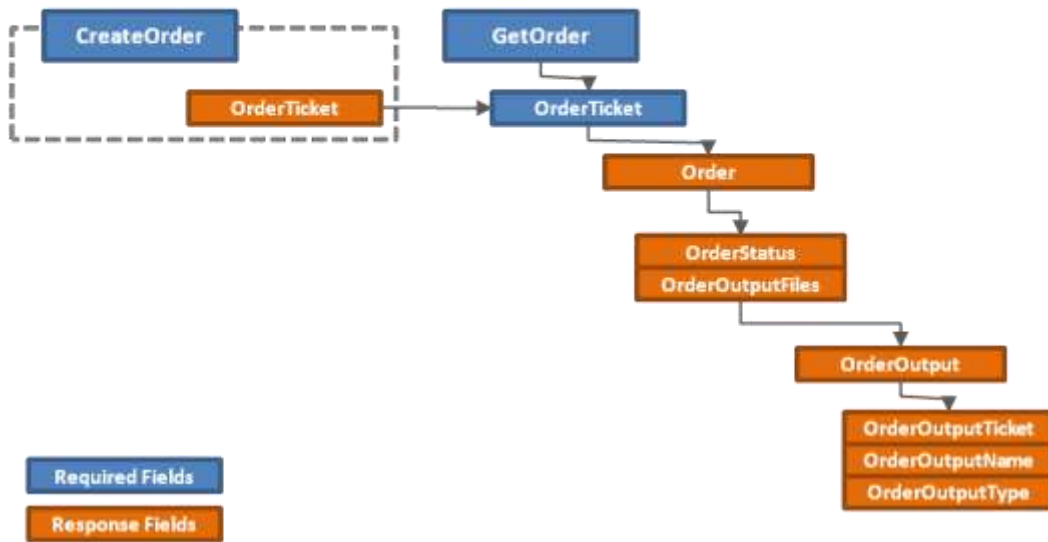
For a simplified purpose of demonstrating the use of the ExecuteOrderRequest, the diagram below visually describes the requested fields and response values of the ExecuteOrder Method: *(This Method is used to begin the processing on an order that has been created and a file has been uploaded to.)*

Image Needs to be updated to reflect the current process with the PaymentProfileService

GetOrder (OrderService - BATCH Process)

For a simplified purpose of demonstrating the use of the GetOrderRequest, the diagram below visually describes the requested fields and response values of the GetOrder Method: *(This Method is most often used to obtain the current order status and obtain the list of files available in the order to output.)*

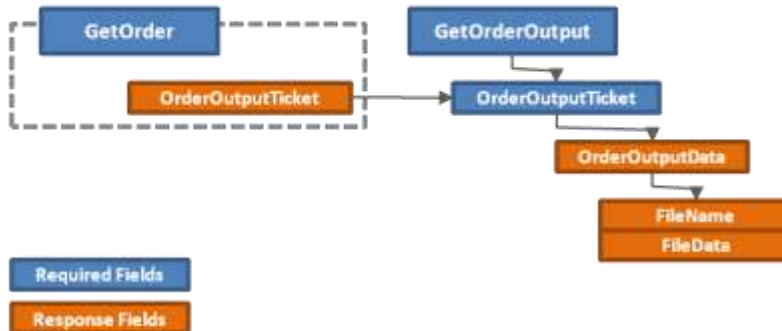
GetOrderRequest -> GetOrderResponse



GetOrderOutput (OrderService - BATCH Process)

For a simplified purpose of demonstrating the use of the GetOrderOutputRequest, the diagram below visually describes the requested fields and response values of the GetOrderOutput Method:

GetOrderOutputRequest -> GetOrderOutputResponse



C# Example of the use of GetOrderOutputRequest and return to local file

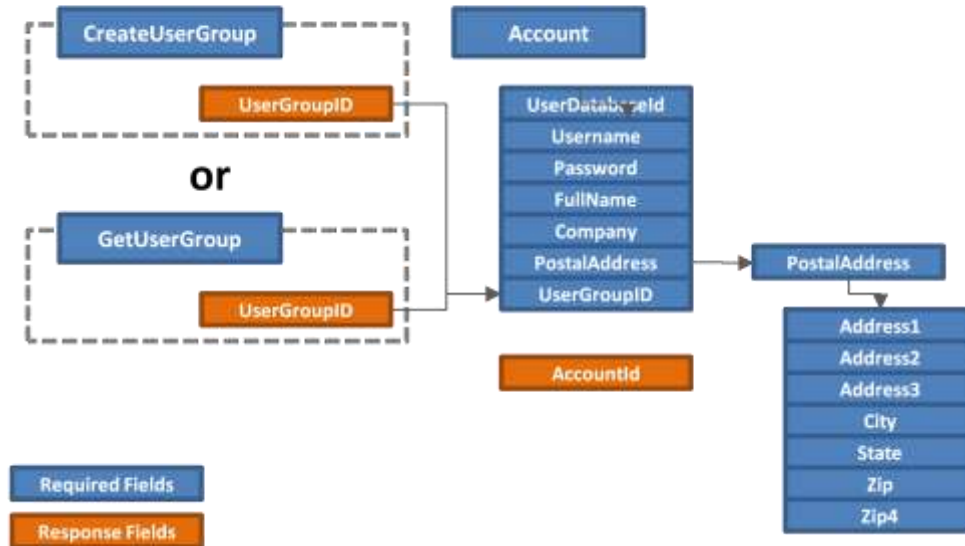
```
GetOrderOutputRequest request = new GetOrderOutputRequest();
request.OrderOutputTicket = outputfileticket;
GetOrderOutputResponse response = orderServiceClient.GetOrderOutput(request);

SaveFileDialog saveFile1 = new SaveFileDialog();
saveFile1.FileName = split.First();
string outputfilepath;
if (saveFile1.ShowDialog() == System.Windows.Forms.DialogResult.OK)
{
    outputfilepath = saveFile1.FileName;
    System.IO.File.WriteAllBytes(outputfilepath, response.OrderOutputData.FileData);
}
```

CreateAccount (AccountService - BATCH Process)

For a simplified purpose of demonstrating the use of the CreateAccountRequest, the diagram below visually describes the requested fields and response values of the CreateAccount Method:

CreateAccountRequest



See next page for C# code example.

C# Example of the use of CreateAccount with a search to determine the existence of UserGroupID

```
//Check to see if user group exists otherwise create one
Guid subclientid = new Guid();
GetUserGroupsRequest usergroups = new GetUserGroupsRequest();
GetUserGroupsResponse usergroupslist = accountServiceClient.GetUserGroups(usergroups);
UserGroup[] users = usergroupslist.UserGroups;

foreach (UserGroup i in users)
{
    if (txtusergroup.Text.Trim() == i.UserGroupName.Trim())
    {
        subclientid = i.UserGroupId;
        break;
    }
}

if (subclientid == new Guid("{00000000-0000-0000-0000-000000000000}"))
{
    CreateUserGroupRequest usergrouprequest = new CreateUserGroupRequest();
    UserGroup usergroup = new UserGroup();
    usergroup.UserGroupName = txtusergroup.Text.Trim();

    usergrouprequest.UserGroup = usergroup;

    CreateUserGroupResponse usergroupresponse = accountServiceClient.CreateUserGroup(usergrouprequest);
    subclientid = usergroupresponse.UserGroup.UserGroupId;
}

//Create the new account

AccountService.Account account = new AccountService.Account();

AccountService.PostalAddress postalAddress = new AccountService.PostalAddress();

account.FullName = txtaccountname.Text;
account.Company = txtaccountcompany.Text;
account.UserGroupId = subclientid;
account.Password = txtaccountpassword.Text;
account.Username = txtaccountuserid.Text;
account.UserDatabaseId = new Guid("{ " + txtaccountuseridbaseid.Text + " }");

postalAddress.AddressLine1 = "804 Liberty Blvd";
postalAddress.AddressLine2 = "Suite 201";
postalAddress.AddressLine3 = "";
postalAddress.City = "Sun Prarie";
postalAddress.State = "WI";
postalAddress.Zip = "53590";
postalAddress.Zip4 = "";

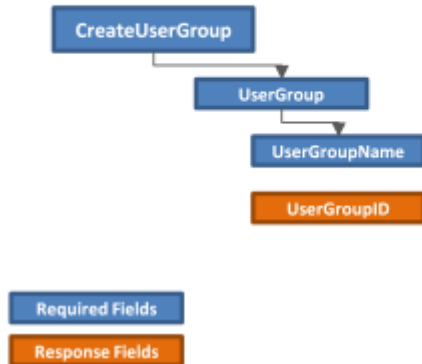
account.PostalAddress = postalAddress;

CreateAccountRequest request = new CreateAccountRequest();
request.Account = account;
CreateAccountResponse response = accountServiceClient.CreateAccount(request);
```

CreateUserGroup (AccountService - BATCH Process)

For a simplified purpose of demonstrating the use of the CreateUserGroupRequest, the diagram below visually describes the requested fields and response values of the CreateUserGroup Method:

CreateUserGroupRequest



C# Example of the use of CreateUserGroup

```
CreateUserGroupRequest usergrouprequest = new CreateUserGroupRequest();
UserGroup usergroup = new UserGroup();
usergroup.UserGroupName = txtusergroup.Text.Trim();

usergrouprequest.UserGroup = usergroup;

CreateUserGroupResponse usergroupresponse = accountServiceClient.CreateUserGroup(usergrouprequest);
subclientid = usergroupresponse.UserGroup.UserGroupID;
```